IBM WebSphere Application Server for z/OS, Version 8.0

Establishing high availability

SA23-1362-00

Note

Before using this information, be sure to read the general information under “Notices” on page 145.

Compilation date: June 9, 2011

Contents

How to send your comments

Your feedback is important in helping to provide the most accurate and highest quality information.

- To send comments on articles in the WebSphere Application Server Information Center
  1. Display the article in your Web browser and scroll to the end of the article.
  2. Click on the **Feedback** link at the bottom of the article, and a separate window containing an e-mail form appears.
  3. Fill out the e-mail form as instructed, and click on **Submit feedback** .
- To send comments on PDF books, you can e-mail your comments to: **wasdoc@us.ibm.com** or fax them to 919-254-5250.

  Be sure to include the document name and number, the WebSphere Application Server version you are using, and, if applicable, the specific page, table, or figure number on which you are commenting.

When you send information to IBM, you grant IBM a nonexclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you.

Changes to serve you more quickly

Print sections directly from the information center navigation

PDF books are provided as a convenience format for easy printing, reading, and offline use. The information center is the official delivery format for IBM WebSphere Application Server documentation. If you use the PDF books primarily for convenient printing, it is now easier to print various parts of the information center as needed, quickly and directly from the information center navigation tree.

To print a section of the information center navigation:

1. Hover your cursor over an entry in the information center navigation until the **Open Quick Menu** icon is displayed beside the entry.
2. Right-click the icon to display a menu for printing or searching your selected section of the navigation tree.
3. If you select **Print this topic and subtopics** from the menu, the selected section is launched in a separate browser window as one HTML file. The HTML file includes each of the topics in the section, with a table of contents at the top.
4. Print the HTML file.

For performance reasons, the number of topics you can print at one time is limited. You are notified if your selection contains too many topics. If the current limit is too restrictive, use the feedback link to suggest a preferable limit. The feedback link is available at the end of most information center pages.

Under construction!

The Information Development Team for IBM WebSphere Application Server is changing its PDF book delivery strategy to respond better to user needs. The intention is to deliver the content to you in PDF format more frequently. During a temporary transition phase, you might experience broken links. During the transition phase, expect the following link behavior:

- Links to Web addresses beginning with http:// work
- Links that refer to specific page numbers within the same PDF book work
- The remaining links will *not* work. You receive an error message when you click them

Thanks for your patience, in the short term, to facilitate the transition to more frequent PDF book updates.

Chapter 1. How do I establish highly available services for applications

Follow these shortcuts to get started quickly with popular tasks.

When you visit a task in the information center, look for the **IBM® Suggests** feature at the bottom of the page. Use it to find available tutorials, demonstrations, presentations, developerWorks® articles, IBM Redbooks®, support documents, and more.

Create a new core group (high availability domain)

Configure the core group bridge service

Set up a high availability sysplex environment

Configure high availability and workload sharing of service integration

Set up high availability for the transaction service

Set up high availability for resource adapters

Chapter 2. Setting up a high availability environment

The high availability framework that is provided with the product eliminates single points of failure and provides peer to peer failover for applications and processes running within the product environment. This infrastructure is managed by a high availability manager and includes cells, clusters, core groups, and high availability groups. Every high availability group has a policy associated with it that the high availability manager uses to determine which members of a high availability group are active at a given time.

Before you begin

Plan out how you need to set up your high availability environment to avoid the risk of a failure without failover coverage. As part of the planning process, understand how the high availability manager can assist you in controlling this type of environment.

About this task

The high availability manager is designed to function in all of the supported product topologies. However, a high availability-managed environment must comply with the following rules:

- A cell in a high availability infrastructure is partitioned into one or more core groups. The product provides a default core group as part of the high availability manager function. You can use the administrative console to create additional core groups.
- A core group cannot extend beyond the boundaries of a cell, and it cannot overlap with any other core groups.
- A cluster must be a member of only one core group. All of the individual members of that cluster must be members of the same core group.
- Individual application servers are also members of a core group.
- All running members of a core group must be able to communicate with all of the other running members of that same core group.

While administering your core groups, you might need to perform one or more of the following tasks. These tasks can be performed in any order.

Procedure

1. Enable the high availability manager if it is not already enabled.
2. View information about the high availability groups, core groups, and core group members.
3. Create a new policy and associate it with a high availability group.
4. Change the policy that is associated with a high availability group.
5. Create a new core group.
6. Change the configuration settings for a core group.

   While running your applications in a highly available environment, you might want to move core group members to a different core group. You also might want to change one or more of the following:

   - The servers that designated as preferred coordinators.
   - The number of core group coordinators.
   - The settings for the discovery protocol, or the failure detection protocol. You might also want to change the version of the protocol.
   - Select an alternate protocol provider to use instead of the default Discovery Protocol and Failure Detection Protocol.
   - The replication settings for a core group.
7. Configure a transport for a core group.

If you are using an IIOP transport, you might also need to complete one or more of the following actions:

- Configure core group IP caching.
- Configure socket buffers for the core group.

8. Specify a core group when you add a node.
9. Specify a core group when you create an application server.
10. Set up a core group bridge.

    If you are using multiple core groups and members of different groups need to communicate with each other, you must set up a core group bridge to enable this communication.
11. Set up a sysplex that is highly available.
12. Control application rollout and workload routing in a high availability configuration.
13. Manually update a high availability application.

What to do next

After you set up your product environment to comply with all of the high availability-managed environment rules, use the default core group to control this environment.

gotcha: Do not add core groups unless you cannot properly perform without them. Also, do not change the default configurations unless doing so provides the solution to a specific problem. When you do make configuration changes, such as changing the policy for a high availability group or moving core group members between core groups, make sure you fully understand the effect such changes have on your entire environment. Such changes might affect application processing.

See the *Troubleshooting and support* PDF if you encounter problems while running applications in your high availability environment.

Chapter 3. High availability manager

The product includes a high availability manager component. The services that the high availability manager provides are only available to product components.

A high availability manager provides several features that allow other product components to make themselves highly available. A high availability manager provides:

- A framework that allows singleton services to make themselves highly available. Examples of singleton services that use this framework include the transaction managers for cluster members, and the default messaging provider, also known as the service integration bus.
- A mechanism that allows servers to easily exchange state data. This mechanism is commonly referred to as the bulletin board.
- A specialized framework for high speed and reliable messaging between processes. This framework is used by the data replication service when the product is configured for memory-to-memory replication.

A high availability manager instance runs on every application server, proxy server, node agent and deployment manager in a cell. A cell can be divided into multiple high availability domains known as core groups. Each high availability manager instance establishes network connectivity with all other high availability manager instances in the same core group, using a specialized, dedicated, and configurable transport channel. The transport channel provides mechanisms which allow the high availability manager instance to detect when other members of the core group start, stop, or fail.

Within a core group, high availability manager instances are elected to coordinate high availability activities. An instance that is elected is known as a core group coordinator. The coordinator is highly available, such that if a process that is serving as a coordinator stops or fails, another instance is elected to assume the coordinator role, without loss of continuity.

Highly available components

A *highly available component* is a component for which a high availability group is defined on the processes where that component can run. The coordinator tracks high availability group membership, and knows on which processes each highly available component can run.

The coordinator also associates a high availability policy with each high availability group. A *high availability policy* is a set of directives that aid the coordinator in managing highly available components. For example, a directive might specify that a component runs on a specific process, if that process is available. Directives are configurable, which makes it possible for you to tailor policies to your installation.

The coordinator is notified as core group processes start, stop or fail and knows which processes are available at any given time. The coordinator uses this information, in conjunction with the high availability group and policy information, to ensure that the component keeps functioning. The coordinator uses the policy directives to determine on which process it starts and runs each component. If the chosen process fails, the coordinator restarts the component on another eligible process. This reduces the recovery time, automates failover, and eliminates the need to start a replacement process.

State data exchange

The high availability manager provides a specialized messaging mechanism that enables processes to exchange information about their current state. Each process sends or posts information related to its current state, and can register to be notified when the state of the other processes changes. The workload management (WLM) component uses this mechanism to build and maintain routing table information. Routing tables built and maintained using this mechanism are highly available.

Replication

The data replication service (DRS) that is provided with the product, is used to replicate HTTP session data, stateful EJB sessions, and dynamic cache information among cluster members. When DRS is configured for memory-to-memory replication, the transport channels defined for the high availability managers are used to pass this data among the cluster members.

When to use a high availability manager

A high availability manager consumes valuable system resources, such as CPU cycles, heap memory, and sockets. These resources are consumed both by the high availability manager and by product components that use the services that the high availability manager provides. The amount of resources that both the high availability manager and these product components consume increases nonlinearly as the size of a core group increases.

For large core groups, the amount of resources that the high availability manager consumes can become significant. Disabling the high availability manager frees these resources. However, before you disable the high availability manager, you should thoroughly investigate the current and future needs of your system to ensure that disabling the high availability manager does not also disable other functions that you use that require the high availability manager. For example, both memory to memory session replication, and remote request dispatcher (RRD) require the high availability manager to be enabled.

The capability to disable the high availability manager is most useful for topologies where none of the high availability manager provided services are used. In certain topologies, only some of the processes use the services that the high availability manager provides. In these topologies, you can disable the high availability manager on a per-process basis, which optimizes the amount of resources that the high availability manager uses.

Do not disable the high availability manager on administrative processes, such as node agents and the deployment manager, unless the high availability manager is disabled on all application server processes in that core group.

Some of the services that the high availability manager provides are cluster based. Therefore, because cluster members must be homogeneous, if you disable the high availability manager on one member of a cluster, you must disable it on all of the other members of that cluster.

When determining if you must leave the high availability manager enabled on a given application server process, consider if the process requires any of the following high availability manager services:

- Memory-to-memory replication
- Singleton failover
- Workload management routing
- On-demand configuration routing

Memory-to-memory replication

Memory-to-memory replication is a cluster-based service that you configure or enable at the application server level. If memory-to-memory replication is enabled on any cluster member, then the high availability manager must be enabled on all of the members of that cluster. Memory-to-memory replication is automatically enabled if:

- Memory-to-memory replication is enabled for web container HTTP sessions. See the *Developing and deploying applications* PDF for more information.
- Cache replication is enabled for the dynamic cache service. See the *Developing and deploying applications* PDF for more information.
- EJB stateful session bean failover is enabled for an application server.See the *Developing and deploying applications* PDF for more information.

Singleton failover

Singleton failover is a cluster-based service. The high availability manager must be enabled on all members of a cluster if one or more instances of the default messaging provider are configured to run in the cluster. The default messaging provider is the messaging engine that is provided with the product.

Workload management routing

Workload management (WLM) propagates the following classes or types of routing information:

- Routing information for enterprise bean IIOP traffic.
- Routing information for the default messaging engine, which is also referred to as the service integration bus.
- Routing HTTP requests through the IBM WebSphere® Application Server proxy server.
- Routing Web Services Addressing requests through the IBM WebSphere Application Server proxy server.
- Routing SIP (Session Initiation Protocol) requests.

WLM uses the high availability manager to both propagate the routing information and make it highly available. Although WLM routing information typically applies to clustered resources, it can also apply to non-clustered resources, such as stand-alone messaging engines. During typical circumstances, you must leave the high availability manager enabled on any application server that produces or consumes either IIOP or messaging engine routing information.

For example, if:

- The routing information producer is an enterprise bean application that resides in cluster 1.
- The routing information consumer is a servlet that resides in cluster 2.

When the servlet in cluster 2 calls the enterprise bean application in cluster 1, the high availability manager must be enabled on all servers in both clusters.

Workload management provides an option to statically build and export route tables to the file system. Use this option to eliminate the dependency on the high availability manager.

On-demand configuration routing

In a WebSphere Application Server, Network Deployment system, the on-demand configuration is used for IBM WebSphere Application Server proxy server routing. If you want to use on-demand configuration routing in conjunction with your web services, you must verify that the high availability manager is enabled on the proxy server and on all of the servers to which the proxy server routes work.

Core groups (high availability domains)

A core group is a high availability domain that consists of a set of processes in the same cell that can directly establish high availability relationships. Highly available components can only fail over to another process in the same core group and replication can occur only between members of the same core group.

A cell must contain at least one core group, although multiple core groups are supported. Each core group contains a core group coordinator to manage its high availability relationships, and a set of high availability policies that are used to manage the highly available components within that core group.

Core group members

Every deployment manager, node agent, application server, and proxy server is a member of a core group. When a process is created it is automatically added to a core group. The core group membership is stored in a product configuration document. You can move processes from one core group to another. The following rules govern the core group membership:

- Every process is a member of exactly one core group.
- All members of a cluster must be members of the same core group.

A core group member has a well-defined life cycle. When the first core group member starts, the transport that is dedicated to that core group automatically starts. The Discovery Protocol, View Synchrony Protocol, and Failure Detection Protocol for that core group member also start and run for the entire lifetime of the core group member:

- The Discovery Protocol is responsible for discovering when other core group processes start, and for opening network connections to these other members.
- The View Synchrony Protocol is responsible for establishing reliable messaging with other core group members after the connections are opened.
- The Failure Detection Protocol is responsible for detecting when other core group members stop or become unreachable because of a network partition.

Core group coordinator

The core group coordinator is responsible for coordinating high availability activities between the core group members for which View Synchrony Protocol is established.

Core group transport

Network communication between all the members of a core group is essential. The network environment must consist of a fast local area network (LAN) with full Internet Protocol (IP) visibility and bidirectional communication between all core group members. Each core group member must be able to receive communications from any of the other core group members.

Multiple core groups

A cell, by default, contains a single core group, called DefaultCoreGroup. All processes in the cell are initially members of this core group. A single core group is usually sufficient. However, some topologies or special circumstances require multiple core groups. There are also topologies that do not require multiple core groups but having them is a good practice. For example, you might want to define multiple core groups if :

- A large number of processes in the cell and the core group protocols, such as the View Synchrony Protocol, consume correspondingly large amounts of resources such as CPU.
- Core group protocols, such as the Failure Detection Protocol, need tuning or configuring to use values that work best with smaller numbers of core group members.

If members of different core groups need to share workload management or on-demand configuration routing information, use the core group bridge service to connect these core groups. The core group bridge service uses access point groups to connect the core groups. A core group access point defines a set of bridge interfaces that resolve to IP addresses and ports. The core group bridge service uses this set of bridge interfaces to enable members of one core group to communicate with members of another core group.

Core group migration considerations

High availability manager and core group functionality is provided in Version 6 and higher. This topic discusses core group configuration and topology considerations that might impact your migration if you are migrating from a version of the product that does not contain this functionality, such as Version 5.1.

Before reading this article, you should understand the basic core group concepts contained in the following topics:

- "High availability manager"
- "When to use a high availability manager"
- "Core groups"
- "Core group transports"
- "Core group coordinator"
- "Core group administration considerations"
- "Core group scaling considerations"

Because special planning and migration activities might be required for your environment, before migrating from a version of the product that does not have a high availability manager to one that does have a high availability manager, you should know the answers to the following questions:

- What is the appropriate core group configuration and topology for a cell after it is fully migrated to Version 7.0?
- In a mixed environment, Is the Version 5.x cell configured to use memory to memory replication? If so, how is the replication environment migrated? How is the replication affected during the migration process?
- Which, if any, JMS provider is used in the Version 5.x cell?
- Which, if any, JMS provider is used in the Version 7.0 cell?
- If the Version 7.0 default IBM messaging provider is to be used in the Version 7.0 cell, should the messaging provider be configured to be made highly available?
- Should transaction log recovery be configured in the Version 7.0 cell?

Default core group related migration activities

Core group related activities that are automatically performed during the migration process are relatively simple and straightforward. When you migrate from a Version 5.x environment to a Version 7.0 environment, the following actions occur in the indicated order:

1. The deployment manager is migrated to Version 7.0.
2. During the migration process, a Version 7.0 deployment manager profile and a core group named DefaultCoreGroup is created.
3. The new deployment manager process is added to DefaultCoreGroup.
4. Version 5.x nodes are migrated one by one to Version 7.0. As each of the nodes is migrated, the node agent and the application servers on the migrating node are added to the DefaultCoreGroup.

When the migration finishes, all of the processes in the Version 5.x cell are members of the Version 7.0 DefaultCoreGroup. Because the DefaultCoreGroup is configured with the default values for all settings, the migration process does not configure preferred coordinator servers and does not partition the coordinator across multiple servers. The migration process also does not create new high availability policies, and does not provide for any custom property configuration overrides.

Planning the Core Group Topology

For most Version 5.x topologies, a default migration yields an appropriate and workable Version 7.0 core group topology. In some cases, you might need to make minor changes to the topology, such as setting a non-default transport or configuring the core group for replication. If the Version 5.x cell is large enough to require multiple core groups in the Version 7.0 topology, then more planning should be done before you start the migration process to prevent application outages from occurring when you make your core group configuration changes..

Migrating a large Version 5.x cell to Version 7.0, where multiple core groups are required, can be a complex task. When the Version 7.0 topology requires multiple core groups, you have a number of options as to how, and when to partition the cell into multiple core groups. The approach you take should be based on such factors as the number of processes in the cell, and requirements for continuous application availability. For example, while the normal recommendation is to keep core groups at around 50 members, the practical limit is somewhat higher than 50. For topologies with a small number of applications installed on high end machines (large CPUs with a lot of memory), you might be able to have core groups of up to 200 members. If there are 150 processes in the cell and application availability is not an issue, one option might be to simply migrate the entire cell to Version 7.0, and then create additional core groups. If application availability is an issue, you should create the additional core groups during the migration process so that you do not have to stop and restart core group members after the migration process completes.

Core Group Size

The most important planning consideration is the size of your core group. By default, there is normally one core group per cell. Because core groups do not scale to large sizes, if your Version 5.x cell is large, you might want to create additional core groups for your Version 7.0 topology. You might also need to set up core group bridges if these multiple core groups need to communicate with each other.

Core Group Transport

If a change is made to the core group transport configuration, all core group members must be restarted before the change goes into affect. Therefore, planning is required to minimize the effect of changing the transport. If the transport for the DefaultCoreGroup is changed, the best time to change it is immediately after migrating the Deployment Manager, since at that point in time only the Deployment Manager will need to be restarted. If other core groups are created, then the transport should be configured properly as the new core groups are created.

Custom Property Configuration Overrides

A number of core group configuration parameters can be changed via Custom Property overrides. The available custom property overrides are documented in other Information Center articles in this section.

Whenever a Custom Property override is added, removed or changed, all core group members must be restarted in order to pick up the change. Therefore, planning is required to minimize the effect of changing Custom Properties. If Custom Properties must be changed for the DefaultCoreGroup is changed, the best time to change it is immediately after migrating the Deployment Manager. If other core groups are created, then the Custom Properties should be changed as the new core groups are created.

Core Group Coordinator

Configuring preferred coordinator servers is a best practice. Since the HA Manager can dynamically reread and apply core group coordinator configuration changes, a restart of all core group members to pick up this change is not required

Example: A Large Cell Migration

The following example illustrates some of the thought processes that you should go through as you plan for and execute the migration of a large Version 5.x cell to Version 7.0, where multiple core groups are required. For the purpose of this example, assume your Version 5.x cell has the following topology characteristics:

- The cell contains eight nodes, that are named Node1, Node2, Node3, ..., Node8, not including the deployment manager node.
- The cell contains ten clusters, that are named Cluster1, Cluster2, Cluster3, ..., Cluster10.

- Clusters Cluster1 through Cluster9 each contains 32 application servers. The cluster members for these clusters are distributed symmetrically, four application servers per node, across all nodes
- Cluster10 contains 28 application servers. Cluster10 does not have any application servers on Node1. The application servers for Cluster10 are distributed symmetrically, four application servers per node, across nodes Node2 through Node8.
- There are a total of 316 application servers, 8 node agents and a deployment manager in the cell.
- Each cluster has an application deployed to it that uses EJBs. and these applications can communicate with each other. Therefore, Work Load Management (WLM) routing information must be available everywhere in the cell.
- Applications must be continuously available during the migration.
- The migration is performed over a period of days or weeks.

The first things to consider in planning the Version 7.0 core group topology is that this cell contains 325 processes, and that continuous availability of applications is a requirement. These factors prevent us from simply migrating the entire cell and then reconfiguring the core groups. You must distribute the processes contained in the cell amongst multiple core groups as part of the migration process.

When determining how you want to distribute the V5.x cell processes amongst the new core group, make sure that each core group adheres to the following core group rules:

- Each core group must contain at least one administrative processes. Because the cell in this example has nine administrative processes, 8 node agents and the deployment manager, the maximum number of core groups possible in this topology is nine.
- All members of a cluster must be members of the same core group.
- The number of processes contained in each core group should not exceed the recommended size of about 50 members.

Following these rules for this example:

- At least one of the core groups must contain two clusters because, you can only split the cell into a maximum of nine core groups, and there are ten clusters in the V5.x cell.
- Any of the core groups that contain multiple clusters, will have more than 50 members because each cluster contains either 28 or 32 application servers,

While the number of members in at least one core groups will exceed the recommended limit, the number of members is well within the practical limit, and should not create a problem.

Because the applications in this example require the WLM routing information for each cluster contained in the cell, core group bridges must be set up to enable communication between all of the core groups. (Refer to the core group bridge topics if you are not familiar with how to set up a core group bridge.) An appropriate core group bridge topology for this example includes:

- A core group access point for each core group. Each access point contains the set of processes that provide the bridge interfaces for the core group. The bridge interfaces are the processes that actually communicate with processes in other core groups.
- Two bridge interfaces for each access point to eliminate the core group bridge as a single point of failure. These two bridge interfaces will also be placed on different nodes to further ensure continual availability.

  When you select processes to serve as the bridge interfaces, remember that bridge interfaces need extra memory and CPU cycles. Normally node agents are good processes to use as bridge interfaces because during normal operations a node agent has a lower workload than an application servers or the deployment manager.

  However, in this example, there are only eight node agents available to serve as bridge interfaces. Because the topology wants two bridge interfaces per access point, if you only use node agents as bridge interfaces, you are limited to four access points, and subsequently four core groups. Therefore, before starting the migration process, you might want to create eight stand-alone servers to specifically act as bridge interfaces, and that do not host applications. Then each access point can contain one node agent and one stand-alone bridge interface server. This setup gives you a total of eight access points and eight core groups.

- A single core group access point group that contains all of the access point. A single core group access point group ensures that all bridge interface processes can communicate directly. These bridge interfaces form a fully connected mesh.

  An alternative topology is to use multiple access point groups, which results in a chain topology. In a chain topology communication is forwarded from one bridge interface to another through intermediate bridge interfaces along the chain.

Now that you have determined the setup for your core group bridge interfaces, you are ready to decide how to distribute the ten clusters, eight node agents, eight stand-alone bridge interface servers, and the deployment manager across your eight core groups. You want to distribute the processes as evenly as possible across the eight core groups. The following topology is a good example of how to evenly distribute the process contained in the V5.x cell:

- The first core group, DefaultCoreGroup, contains the deployment manager, the node agent from Node1, the bridge server from Node 2 and Cluster1.
- Core Group 2 contains the node agent from Node2, the bridge server from Node3 and Cluster2
- Core Group 3 contains the node agent from Node3, the bridge server from Node4 and Cluster3

The default transport in this example does not need to change.

Because this example does not indicate that you will need more than one coordinator per core group, you can leave the coordinator setting at the default value of 1. However, you might want to make the stand-alone bridge interface server, that is contained in each core group, the preferred coordinator server for that core group. This designation initially keeps the work load required of a coordinator away from the clustered application servers that are running applications.

Your migration plan

If, after reviewing the preceding example and completing the initial planning process for the cell you are migrating, you determine that the default migration flow is not appropriate for your target Version 7.0 topology, it is time to develop a plan or a road map for the actual migration process. This plan should include all necessary extra core group related steps for migrating from Version 5.x to Version 7.0. and answers to the following questions:

When will you create the new core groups?

The best time to create the new core groups is immediately after the deployment manager migration completes. As the new core groups are created, you should configure the previously mentioned custom properties You can use either administrative console or the `createCoreGroup` wsadmin command to create your new core groups. However, you must use the administrative console to configure the custom properties.

What actions do you need to perform as nodes are migrated?

As each node is migrated, you should:

- Create the new stand-alone application server that is to be one of your core group bridge interfaces.
- Adjust the transport buffer size on all processes on the node. A script is the best option for performing this action.
- Adjust the heap size on the node agent and the stand-alone server, and turn on verbose GC for these processes.

All of these changes must be completed before you restart the migrated node. You can use the administrative console to make these, and then perform a manual synchronization of the nodes configuration before restarting the node agent and application servers.

When and how are processes moved to new core groups?

By default, the migration process places all processes in the core group named DefaultCoreGroup. At some point in time the number of members contained in this core group will exceed the size limits and you must redistribute the processes to other core groups. It is important to understand that the processes must be stopped before they can be moved. If continuous application availability is required, you must carefully plan out the order in which you will move the processes to different core groups.

You can use either the administrative console or the `moveServerToCoreGroup` wsadmin command to move the deployment manager, node agents and stand-alone application server.

Moving clustered application servers is more complicated. Under normal circumstances, You can use either the administrative console or the `moveServerToCoreGroup` wsadmin command to move clusters. However, during the migration process, because the cluster to be moved might have both Version 7.0 and Version 5.x members, the normal commands fail because a Version 5.x cluster member is not yet a member of any core group. To move a mixed cluster to a new core group, you must use the `moveClusterToCoreGroup` wsadmin command with the optional `checkConfig` parameter.

For example, suppose Cluster0 has cluster members A, B, C and D. Member A is on a node that has been migrated to Version 7.0 and is a member of the DefaultCoreGroup, while B, C and D are still on Version 5.x nodes. To move Cluster0 to core group CG1 use the following command”

```
$AdminTask moveClusterToCoreGroup {-source CoreGroup1 –target CG1
     –clusterName Cluster0 –checkConfig false}
```

When a clustered application server is migrated, the migration utilities determine if other cluster members have already been migrated and automatically place the migrating member in the same core group as other members of the same cluster that are already migrated.

In the example above, member A was moved to core group CG1. When the nodes containing B, C and D are migrated, migration will place these cluster members in CG1 instead of the DefaultCoreGroup. Therefore, it is necessary to run the moveClusterToCoreGroup command only once for each cluster.

When do you need to configure your core group bridges?

By the time you move your processes to multiple core groups, you must have core group bridges configured and running. This means that the processes that you want to use as bridge interfaces in your Version 7.0 target topology might not be available when they are initially needed because they have not been migrated from the Version 5.x nodes. Therefore, to ensure continual availability of your applications, you must configure some clustered application servers to be temporary bridge interfaces while the migration continues. After all of the processes have been migrated to Version 7.0, you can adjust the core group bridge configuration to match your desired Version 7.0 topology.

Other planning considerations

If your target Version 7.0 configuration requires multiple core group bridges, use the IBM_CS_WIRE_FORMAT_VERSION core group custom property to implement scaling improvements.

Also, if all of your core groups are bridged together and routing shared information amongst each other, the amount of data shared between the core group members is likely to be much larger than normal. Therefore, you should use the following settings to increase the core group memory settings to allow for a more efficient transfer of data:

- Set the IBM_CS_DATASTACK_MEG to 100
- Set the transport buffer size on all processes to 100.

You should also consider adjusting such factors such as JVM heap sizes for any node agent or application server that is being used as a bridge interface, and any stand-alone server that is being used as a coordinator. A recommended starting point is to increase the heap size by 512 megabytes. You can also turn on verbose GC monitoring for these processes so that you can fine tune these heap sizes over time.

Possible migration flows

There are a number of migration flows that you can implement for a successful migration. The following flows assume a common starting point where the deployment manager migration has completed and the core groups have been created, but no further actions have been taken.

Migration Flow 1

In this flow, we strictly follow the rules. This flow is unsatisfactory for a number of reasons. As each node is migrated, clusters will need to be moved. This requires stopping all cluster members. This may lead to applications not being available. In addition, the bridges need to be reconfigured at each step.

- Migrate Node1. The DefaultCoreGroup contains the deployment manager and all the processes from Node1. Since the DefaultCoreGroup contains less than 50 members, no further action is required.
- Migrate Node2. The DefaultCoreGroup now contains more than the recommended number of processes. Balance the processes over 2 core groups by moving half of the clusters and the node agent for Node2 into CoreGroup2. Since there are now multiple core groups being used, we need to configure the core group bridge. Create bridge interface servers on nodes Node1 and Node2. Configure the core group bridge to bridge the two core groups together.
- Migrate Node3. Balance the processes across 3 core groups by moving some of the clusters from the DefaultCoreGroup and CoreGroup2 to CoreGroup3. Move the node agent for Node3 to CoreGroup3. Create the bridge interface server on Node3. Reconfigure the core group bridge to bridge all three core groups together.
- Continue migrating the nodes until the migration is complete. As each node is migrated, some rebalancing and reconfiguring of the core group bridge may be necessary.

Migration Flow 2

In this flow, we temporarily bend the rules. This flow yields better results, as running application servers do not need to be stopped to move them to a different core group. While the migration is in progress, some core groups will not contain an administrative process for some period of time. This is a technical violation of the rules, but is acceptable as long as the core group configuration is not changed while the migration is in progress.

- Migrate Node1. Node1 contains members from all clusters except Cluster10
- Move all possible clusters to core group identified in the final target topology. The deployment manager, node agent for Node1 and Cluster1 are already in the DefaultCoreGroup, so no further action is required for them. Move Cluster2 to CoreGroup2, Cluster3 to CoreGroup3 and so on. Create the bridge server for Node1 and place it in CoreGroup2.
- Configure the core group bridge to bridge all 8 core groups together. For simplicity we will temporarily configure a single bridge interface for each access point. (This will introduce a single point of failure while the migration is in progress) Since most of the bridge interfaces from the final topology are still on Version 5.x, we need to use application servers as temporary bridge interfaces in 6 of the 8 core groups. This may require a temporary increase in the heap size of selected application servers.
- Migrate Node2. Migration will automatically move the clustered application servers for Node2 to the proper core groups. Since Cluster10 did not have any application servers on Node1, manually move Cluster10 to CoreGroup8. Move the node agent for Node2 to CoreGroup2. Create the bridge server on Node2. Optionally, reconfigure the core group bridge so that some of the temporary bridge interface servers are on Node2 to help spread the load across both nodes.
- Continue migrating the nodes using the same pattern until all nodes have been migrated.

- When all nodes have been migrated, configure preferred coordinator servers. Reconfigure the bridge interfaces to match the final target topology (with two bridge interface servers in each access point) Stop and restart the servers that are serving as temporary bridge interfaces. Restart the new bridge interface servers.

Migration Flow 3

This flow is a variation on Flow 2. As noted, this flow is a variation on Flow 2. The benefit is that the initial bridge load is spread across three nodes instead of 1. The disadvantage is that the initial redistribution of clusters to core groups occurs after Node3 has migrated. This requires that the servers running on nodes Node1 and Node2 must be stopped in order for the move to occur. This may affect application availability.

- Migrate Node1. When this step is complete, the DefaultCoreGroup will contain 38 processes, which is within limits.
- Migrate Node2. When this step is complete, the DefaultCoreGroup will contain 79 processes. While this is larger than the recommended size, it is well within the practical limit.
- Migrate Node3. Move all clusters to core group identified in the final topology. Move Cluster2 to CoreGroup2, Cluster3 to CoreGroup3 and so on. Move the three node agents to the proper core groups. Create and move the three bridge interface servers to the proper core groups.
- Select clustered application servers to act as temporary bridges for the core groups that do not yet contain designated bridge interfaces. Temporarily adjust the heap sizes on these servers. Configure the core group bridge to bridge all 8 core groups together.
- Continue migrating the nodes until all nodes have been migrated.
- When all nodes have been migrated, configure preferred coordinators. Reconfigure the bridge interfaces to match the final target topology. Stop and restart processes as required.

Core group coordinator

Every core group has a coordinator that manages high availability activities between the core group members. The coordinator manages the failover of highly available singleton services and distributes live server state data to interested core group members. The coordinator uses some CPU and memory (JVM heap) resources to perform these tasks. In some configurations, the amount of resources that the coordinator uses might be large.

Note: This topic references one or more of the application server log files. Beginning in WebSphere Application Server Version 8.0 you can configure the server to use the High Performance Extensible Logging (HPEL) log and trace infrastructure instead of using `SystemOut.log` , `SystemErr.log`, `trace.log`, and `activity.log` files or native z/OS logging facilities. If you are using HPEL, you can access all of your log and trace information using the LogViewer command-line tool from your server profile bin directory. See the information about using HPEL to troubleshoot applications for more information on using HPEL.

The coordinator workload can be divided over multiple coordinator instances. Each instance runs on a different core group member and is assigned a portion of the overall coordination workload. Dividing the workload across multiple coordinator instances enables you to share the associated resource costs across machines. The coordinator function remains highly available, regardless of how its workload is divided or assigned to core group members.

Coordinator election

When a core group member starts or stops, the View Synchrony Protocol installs a new view. The view consists of the core group members that are connected and cooperating. Whenever a new view is installed, it might be necessary to redivide the coordinator workload among the core group members. For example, a core group member that is hosting a coordinator instance might fail and the high availability manager must elect a replacement coordinator.

Informational messages, similar to the following message, are logged in the `SystemOut.log` file when a particular core group member is elected as a coordinator:

```
HMGR0206I: The coordinator is an active coordinator for core group DefaultCoreGroup
```

Messages, similar to the following message, are logged if a core group member is no longer an elected coordinator:

```
HMGR0207I: The coordinator was previously an active coordinator for core group
           DefaultCoreGroup but has lost leadership.
```

gotcha: Remember that coordinator election occurs whenever the view changes. Electing a new coordinator uses a lot of resources because this process causes increased network traffic and CPU consumption. Specifying a preferred coordinator server, whenever practical, helps eliminate the need to make frequent coordinator changes.

Multiple coordinators

The core group configuration data contains a field in which users can specify the number of coordinators. The default value for this field is `1`. This default value is sufficient for most installations and applications. Use multiple coordinators when the core group member that is selected as the coordinator uses noticeably more memory or CPU than similar core group members. In addition, some software products that heavily use the high availability framework instruct you to increase the number of coordinators.

Preferred servers

When you configure a core group, you can specify the core group members the high availability manager should use as coordinators, if they are available. Preferred coordinator servers should be core group processes that are cycled as infrequently as possible. The preferred coordinator servers should also be hosted on machines with excess capacity.

Specifying preferred coordinator servers is a good practice. When coordinators are elected during a view change, the high availability manager checks for a list of preferred servers. If there is a list, the high availability manager selects a server from that list as the coordinator. If there is no list, the high availability manager selects the view member with the lexically lowest name as the coordinator, which incurs some overhead if it causes the coordinator to move.

Core group administration considerations

Core group configuration information is stored in a CoreGroup configuration object that is backed by a `coregroup.xml` document. Process-specific configuration information for each core group member is stored in a HAManagerService configuration object that is backed by a `hamanagerservice.xml` document.

The `coregroup.xml` document is a cell-scoped document. The master copy of this document is stored in the configuration repository for the deployment manager. A copy of this document is shadowed to every node in the cell. The `coregroup.xml` document includes of the following configuration information:

- The list of core group members
- The high availability policies for the core group
- The core group coordinator configuration information
- The core group transport configuration information, including memory buffer size settings
- Discovery and failure detection protocol configuration settings

The core group member process-specific configuration information stored in the `hamanagerservice.xml` document includes:

- Whether the high availability manager is enabled.
- The name of the core group to which the member belongs.
- How frequently the high availability manager checks the health of highly available singletons running on the member, if a length of time is in affect for this function.

Core group configuration document

transition: The requirement that every core group must contain at least one administrative process, that exists in Versions 6.0.x and 6.1.x, does not apply to for this version of the product. However, if you are running in a mixed cell environment, every core group that contains any Version 6.x members must also contain an administrative process.

The master copy of the core group configuration document is directly modified when direct attributes, such as the coordinator configuration, are modified. The master copy of the core group configuration document is implicitly modified when a server is created or deleted, or a node is added or removed. In either case, the list of core group members is updated to reflect which processes are added or removed.

The set of core group members for which the View Synchrony Protocol is established is commonly referred to as a *view*. Whenever a view is installed, one of the core group members is elected to send its current configuration to all other members of the view. This processing ensures that all members of the view are running with a consistent core group configuration. This processing also means that inconsistencies in a high availability policy or coordinator configuration are tolerated. However, inconsistencies in the list of core group members or the core group transport are not tolerated.

If you modify a list of core group members, do not start a member of that core group until you are sure that the change is fully synchronized to all nodes in the cell. If a node agent is down when the configuration change is made, you must manually synchronize the configuration change before any processes are started on that node. If you do not manually synchronize the change, the process that is starting cannot establish the View Synchrony Protocol with the other core group members because when a core group member starts, it reads the core group configuration information from the repository on the local node. It then opens connections to other core group members and attempts to establish the View Synchrony Protocol with them. If the local copy of the `coregroup.xml` document is not synchronized with the master core group configuration document, problems occur. For example, if the running processes dynamically reloaded the updated configuration, the configuration for the process that just started is out of sync with the configurations of the other core group members. If the update changed the list of core group members, the list is now inconsistent across the nodes in the cell, and any attempt to establish view synchrony fails because of these inconsistent member lists. When this condition is detected, an error message similar to the following message is logged:

```
DCSV8022I: DCS Stack {0} at Member {1}: Inconsistency of configured defined set
with that of another member. Inconsistent member is {2}. The list of members only
in the local defined set is {3}, whereas the list of members only in the defined
set at the inconsistent member is {4}.
```

When a process detects an inconsistent core group membership condition, the process attempts to reread the core group configuration several times. It is possible that the configuration document is in the process of being synchronized to the node. In such a case, rereading the configuration document can resolve the inconsistency. However, if the process can not resolve the inconsistency after trying to reread the configuration several times, the process stops trying to resolve the inconsistency. To recover from this situation, you must resynchronize the configuration and restart the process.

Core group process-specific configuration document

Unlike the cell-scoped core group configuration information that is contained in the `coregroup.xml` document, the process-specific configuration information for each core group member that is contained in the `hamanagerservice.xml` document cannot be dynamically reloaded. You must restart a process before core group process-specific configuration changes go into affect.

Core group scaling considerations

The amount of system resources, such as CPU and memory, that the high availability manager consumes does not increase linearly as the size of a core group increases. For example, the View Synchrony

Protocol that the high availability manager uses requires a large amount of these resources to maintain a tight coupling over the core group members. Therefore, the amount of resources that a large core group consumes might become significant.

View Synchrony Protocol resource requirements can vary considerably for different core groups of the same size. The amount of resources that the View Synchrony Protocol uses for a core group is determined by:

- The number of applications that are running.
- The type of applications that are running.
- The high availability manager services that are used.

When setting up core group scalability, you must ensure that:

- All of the processes within the cell are distributed properly into core groups of appropriate sizes. Properly distributing these processes limits the amount of resources that the View Synchrony Protocol consumes.
- All of the processes within a given core group are properly configured to support the high availability services that are used within the core group.

Consider implementing one or more of the following scalability techniques to scale the high availability manager in large cells, even if your system is operating properly. The two most basic techniques are:

- Disabling the high availability manager if it is not required.
- Distributing the processes over a number of core groups and using a core group bridge to connect the core groups as required.

Adjusting the size of a core group

Core group size directly effects three aspects of high availability manager processing that impact resource usage:

- The first and most significant aspect is the establishment of the View Synchrony Protocol over a set of active core group members. This activity is commonly referred to as a *view change*.
- The second aspect is the regularly scheduled discovery and failure detection tasks that high availability manager runs in the background.
- The third aspect is the resource usage that results when other product components use high availability manager-provided services.

View Changes

The View Synchrony Protocol creates a new view whenever it detects that there is a change in core group members that are active. A view change typically occurs whenever a core group member starts or stops. When a core group member starts, it opens a connection to all of the other running core group members. When a core group member stops, other core group members detect that their open connections to the stopped member are closed. In either case, the View Synchrony Protocol needs to account for this change. In the case of a newly started member, the View Synchrony Protocol must establish a view that includes the new member. In the case of a stopped member, the View Synchrony Protocol must establish new view for the surviving core group members that excludes the stopped member.

Establishing a new view is an important activity but uses a lot of system resources, especially for large core groups.

- Each running core group member must communicate its current state to other core group members, including information about the messages it has sent or received in the current view.
- All messages sent in a given view must be received and acknowledged by all recipients before a new view can be installed. Under normal operating conditions, receipt of these messages is acknowledged slowly. Completing messages at a view change boundary in a timely fashion requires aggressive acknowledgement and retransmission.

- All core group members must transmit data regarding their current state, such as the set of other core group members to which they can actively communicate.

As the number of active members grows, installing a new view requires a larger, temporary nonlinear increase in high availability manager CPU usage. It is significantly more expensive to add or remove a single member when 50 other core group members exist, than it is to add or remove a member when 20 other members exist.

Installing a new view also triggers state changes in the product components that use the high availability manager. For example, routing tables might need to be updated to reflect the started or stopped member, or a singleton service might need to be restarted on a new member.

The end result is that installing a new view results in a significant, transient spike in CPU usage. If core group sizes become too large, degenerate network timing conditions occur at the view change boundary. These conditions usually result in a failure during an attempt to install a new view. Recovery from such a failure is also CPU intensive. When insufficient CPU is available, or paging occurs, failures can quickly multiply.

Background tasks

The high availability manager periodically runs a number of background tasks, such as checking the health of highly available singleton services that it is managing. Most of these background tasks consume trivial amounts of CPU. The exceptions are the regularly scheduled discovery and Failure Detection Protocols.

The Discovery Protocol attempts to establish communications among core group members that are not currently connected, including processes that are not running. For a given core group that contains N core group members, of which M are currently running, each discovery period results in roughly M x (N – M) discovery messages. Therefore, creating a large number of processes that never start adversely affects the Discovery Protocol CPU usage.

Similarly, when the Failure Detection Protocol runs, each core group member sends heartbeats to all of its established connections to other core group members. For M active members, M x (M-1) heartbeat messages are sent. If aggressive failure detection is required, the size of the core group can adversely affect the amount of CPU usage that heartbeating between core group members consumes.

Smaller core groups positively affect the amount of CPU usage these two protocols consume. For example, if a core group contains 100 active members, 9900 heartbeat messages are sent during every failure detection period. Splitting the 100 member core group into five smaller core groups of 20 members reduces this number of message to 1900, which is a significant reduction.

External usage

Other product components, such as work load management (WLM), and on demand configuration, use high availability manager-provided services, such as live server state exchange, to maintain routing information. The amount of CPU usage that these components consume is linked to core group size. For example, the usage of the live server state exchange to build highly available routing information is linked to the size of the core group.

Distributing processes among multiple core groups

You can use two basic techniques to minimize the amount of resources that the view synchrony and related protocols consume:

- You can disable the high availability manager on processes where the services that the high availability manager provides are not used.
- You can keep core group sizes small.

The key to limiting the high availability manager CPU usage is to limit the size of the core group. Multiple small core groups are much better than one large core group. If you have large cells, create multiple core groups.

The hardware on which you are running the product is also a factor in determining the core group size that is appropriate for your environment.

The current recommendation is to limit core group sizes to 50 members or so. Split large core groups into multiple, smaller core groups. If the resulting core groups need to share routing information, you can use core group bridges to bridge the core groups together.

Adjusting individual core groups based on the application mix and services used

You might need to further adjust Individual core groups based on the application mix and the high availability services that the core group members use.

- Adjust how frequently the default Discovery Protocol and the default Failure Detection Protocol run if the default settings are not appropriate.
- Configure the core group coordinator to run on a specific process or set of processes.
- Partition the coordinator across multiple instances if the consumption of resources by the coordinator process is noticeable.
- Configure the amount of memory that is available to the distribution and consistency services (DCS) and removable media manager (RMM) components for sending network messages when congestion is detected. Congestion can occur under some conditions, even though memory-to-memory replication is not used.

Adjusting ephemeral port ranges

The number of sockets that a core group uses is usually not a major concern. Each core group member must establish a connection with every other member of that core group. Therefore, the number of connections grows exponentially (n-squared) because each connection requires two sockets, one on each end of the connection. Because multiple machines are typically involved, normally you do not have to be concerned about the number of sockets that a core group uses. However, if you have an abnormally large number of core group members that are running on a single machine, you might have to adjust the operating system parameters that are related to ephemeral port ranges. Most operating systems have different default behavior for ephemeral port ranges.

Note: Although it is not an incorrect configuration, if you define ports in the ephemeral range, you might encounter unpredictable behavior. Each operating system has its own ephemeral port range, which you can configure on the operating system level. Due to individual operating system configurations, it is not possible to document specific port numbers.

Core group View Synchrony Protocol

The View Synchrony Protocol is established over the set of core group members that can communicate with each other. This protocol provides guaranteed, in-order message delivery for message streams that involve one sender and potentially multiple receivers. This guarantee is similar to the guarantees that TCP/IP provides for point-to-point message streams.

The set of core group members for which the View Synchrony Protocol is established is commonly referred to as a *view*. Views are unique in time and space. The act of adding or removing members from the view is called a *view change*. A view change is an important and relatively expensive synchronization point. It is also the point where synchronization, consistency and network issues are detected.

The View Synchrony Protocol is transparent to both components using the high availability manager framework and product administrators. However, disruptions in the View Synchrony Protocol might become visible, most notably when a boundary condition known as a view change occurs.

View changes

When a core group member starts, the core group transport and the associated default Discovery Protocol, default Failure Detection Protocol, and View Synchrony Protocol also start. The View Synchrony Protocol establishes an initial view that contains only the local member. The View Synchrony Protocol is notified when the default Discovery Protocol establishes connections with other core group members. The view synchrony layer of the newly connected members then exchange state information. This information is used to determine if a new view can be formed. For example, if a newly started member discovers an existing view, it negotiates with the members of the existing view to establish a new view.

When a member of an established view stops or fails, the default Failure Detection Protocol on the surviving view members detects the failure and notifies the View Synchrony Protocol. The surviving members then establish a new view that excludes the failed member.

Before a new view is established, activities that are related to the current view must be completed. All messages that are sent in the current view must be received and acknowledged by all intended recipients that are still alive. The current members must exchange a non-trivial amount of state information regarding messages sent and received. These members then perform the activities that are needed to complete the pending message activity, which might include the retransmission of messages that seem to be lost.

Installing a new view might result in significant, temporary spikes in the amount of CPU consumed and the network bandwidth used.

View change messages

A view change is a complex multipart procedure and a number of messages are logged every time a view is changed. These messages indicate the stage of view change processing that is complete or is currently running.

For example, the following message indicates that a set of core group members agreed to establish a new view and initiated the view change procedure:

```
DCSV8054I: DCS Stack DefaultCoreGroup at Member
anzioCell01\anzioCellManager01\dmgr: View change in process.
```

The following message indicates that all messages sent in the current view are completed and acknowledged:

```
DCSV2004I: DCS Stack DefaultCoreGroup at Member
anzioCell01\anzioCellManager01\dmgr: The synchronization procedure completed
successfully. The View Identifier is (2:0.anzioCell01\anzioCellManager01\dmgr).
The internal details are [0].
```

The following messages indicate that the view change completed successfully. They also specify the name or identifier for the new view, and the number of core group members in that view:

```
HMGR0218I: A new core group view has been installed. The core group is
DefaultCoreGroup. The view identifier is (3:0.anzioCell01\anzioCellManager01\dmgr).
The number of members in the new view is 2.

DCSV1033I: DCS Stack DefaultCoreGroup at Member
anzioCell01\anzioCellManager01\dmgr: Confirmed all new view members in view
identifier (3:0.anzioCell01\anzioCellManager01\dmgr). View channel type is View|Ptp.
```

The following message provides extended status regarding the state of connections and view synchrony:

```
DCSV8050I: DCS Stack DefaultCoreGroup at Member
anzioCell01\anzioCellManager01\dmgr: New view installed, identifier
(3:0.anzioCell01\anzioCellManager01\dmgr), view size is 2 (AV=2, CD=2, CN=2, DF=6)
```

In this message:

- AV is the number of core group members in the view.
- CN is the number of core group members to which this member has open connections. Normally this number is the same as the number that is specified for AV.
- CD is the number of core group members to which this member has open connections minus the number of bad members. A bad member is one that is connected to this member, but cannot currently establish a view with this member.
- DF is the number of members defined in the core group.

Core group discovery and failure detection protocols

When a core group member starts, no connections to other core group members exist. If a core group is configured to run with either the default Discovery and Failure Detection Protocols or an alternative protocol provider, either the discovery and failure detection tasks or the alternate protocol provider tasks start as part of the process startup procedure. These tasks establish connectivity to other core group members, monitor this connectivity and handle connectivity failures for this core group member, at regularly scheduled intervals, as long as the core group member is active.

The default Discovery Protocol

Note: This topic references one or more of the application server log files. Beginning in WebSphere Application Server Version 8.0 you can configure the server to use the High Performance Extensible Logging (HPEL) log and trace infrastructure instead of using `SystemOut.log` , `SystemErr.log`, `trace.log`, and `activity.log` files or native z/OS logging facilities. If you are using HPEL, you can access all of your log and trace information using the LogViewer command-line tool from your server profile bin directory. See the information about using HPEL to troubleshoot applications for more information on using HPEL.

Note: This topic references one or more of the application server log files. Beginning in WebSphere Application Server Version 8.0 you can configure the server to use the High Performance Extensible Logging (HPEL) log and trace infrastructure instead of using `SystemOut.log` , `SystemErr.log`, `trace.log`, and `activity.log` files or native z/OS logging facilities. If you are using HPEL, you can access all of your log and trace information using the LogViewer command-line tool from your server profile bin directory. See the information about using HPEL to troubleshoot applications for more information on using HPEL.

The default Discovery Protocol establishes network connectivity with the other members of the core group. To establish this connectivity, the Discovery Protocol retrieves the list of core group members and the associated network information from the product configuration settings. The Discovery Protocol then attempts to open network connections to all of the other core group members. At periodic intervals, the Discovery Protocol recalculates the set of unconnected members and attempts to open connections to those members.

When a connection is made to another core group member, the Discovery Protocol notifies the View Synchrony Protocol, and logs this event as an informational message, similar to the following message, in the `SystemOut.log` file.

```
DCSV1032I: DCS Stack DefaultCoreGroup at Member MyCell\anzio\nodeagent:
Connected a defined member MyCell\anzioCellManager\dmgr.
```

Connections can fail at any time for a variety of reasons. The Failure Detection Protocol detects connection failures and notifies the Discovery Protocol. The Discovery Protocol then attempts to open a new network connection to that member at the next scheduled interval.

The amount of CPU cycles that the Discovery Protocol task consumes is proportional to the number of core group members that are stopped or unreachable. The CPU cycles that the Discovery Protocol task consumes is negligible at the default settings.

Default Failure Detection Protocol

The Failure Detection Protocol monitors the core group network connections that the Discovery Protocol establishes. When the Failure Detection Protocol detects a failed network connection, it reports the failure to the View Synchrony Protocol and the Discovery Protocol. The View Synchrony Protocol adjusts the view to exclude the failed member. The Discovery Protocol attempts to reestablish a network connection with the failed member. This task runs as long as the member is active.

The Failure Detection Protocol uses two distinct mechanisms to find failed members:

It looks for connections that closed because the underlying socket was closed.

When a core group member normally stops in response to an administration command, the core group transport for that member also stops, and the socket that is associated with the transport closes. If a core group member terminates abnormally, the underlying operating system normally closes the sockets that the process opened and the socket associated with the core group transport. is closed.

For either type of termination, core group members that have an open connection to the terminated member are notified that the connection is no longer usable. The core group member that receives the socket closed notification considers the terminated member a failed member.

When a failed member is detected because of the socket closing mechanism, one or more of the following messages are logged in the `SystemOut.log` file for the surviving members:

```
DCSV1113W: DCS Stack DefaultCoreGroup at Member anzioCell01\anzioCellManager01\dmgr:
Suspected another member because the outgoing connection to the other member was closed.
Suspected member is anzioCell01\nettuno\ServerB. DCS logical channel is View|Ptp.

DCSV1111W: DCS Stack DefaultCoreGroup at Member anzioCell01\anzioCellManager01\dmgr:
Suspected another member because the outgoing connection from the other member was closed.
Suspected members is anzioCell01\nettuno\ServerB. DCS logical channel is Connected|Ptp.
```

The closed socket mechanism is the way that failed members are typically discovered. TCP settings in the underlying operating system, such as FIN_WAIT, affect how quickly socket closing events are received.

It listens for active heartbeats from the core group members.

The active heart beating mechanism is analogous to the TCP keep alive function. At regularly scheduled intervals, each core group member sends a ping packet on every open core group connection. The rate or periodicity at which the packet is sent is called the heartbeat transmission period.

Each core group member expects to receive a packet on each open connection from the core group member on the other end of the connection. If no packets are received over an open connection within the time length specified for the heartbeat timeout period, then the member on the other end of the connection is marked as failed.

The heartbeat timeout period must be a whole number that is a multiple of the heartbeat transmission period. The heartbeat timeout period must also be at least twice as large as the heartbeat transmission period.

When a member is marked as failed, the following message is sent to the error log file:

```
DCSV1112W: DCS Stack DefaultCoreGroup at Member anzioCell01\anzioCellManager01\dmgr:
Suspected member anzioCell01\nettuno\ServerB because of heartbeat timeout.
Configured Timeout is 180000 milliseconds. DCS logical channel is Connected|Ptp.
```

Active heartbeats are most useful for detecting core group members that are unreachable because the network is stopped. Active heartbeats consume some CPU usage. The amount of CPU usage that is consumed is proportional to the number of active members in the core group. The default configuration for active heartbeats is a balance of CPU usage and timely failed member detection.

You can use the administrative console or the wsadmin tool to configure the heartbeat transmission period and heartbeat timeout period. Read the topic *Configuring the Failure Detection Protocol for a core group* for a description of how to use the administrative console to change these settings.

Alternative protocol providers

You can use an alternate protocol provider instead of the default Discovery Protocol and Failure Detection Protocol to monitor and manage communication between core group members. In general, alternate protocol providers, such as the z/OS® Cross-system Coupling Facility (XCF)-based provider, uses less system resources than the default Discovery Protocol and Failure Detection Protocol, especially during times when the core group members are idle. An alternate protocol provider generally use less system resources because it does not perform the member-to-member TCP/IP pinging that the default protocol providers use to determine if a core group member is still active.

If you decide to use the z/OS Cross-system Coupling Facility (XCF)-based protocol provider, you should understand that at startup, the server process is joined, as a member, to an XCF group. The XCF group contains all of the active members for the core group. XCF provides notification to all of the members of this group whenever a member joins the group, and whenever a member can no longer be contacted because the server shutdown, or XCF determines that the server process has terminated. Whenever a connection between core group members is established, the z/OS Cross-system Coupling Facility (XCF)-based protocol provider notifies the View Synchrony Protocol, and logs this event as an informational message, similar to the following message, in the `SystemOut.log` file.

```
DCSV1032I: DCS Stack DefaultCoreGroup at Member MyCell\anzio\nodeagent:
Connected a defined member MyCell\anzioCellManager\dmgr.
```

Before reconfiguring a specific core group to use an alternative protocol provider, you must verify that the core group meets the following requirements. If the core group does not meet all of these requirements, you must continue to use the default Discovery Protocol and the default Failure Detection Protocol with this core group.

- The core group is homogenous. This means that the core group processes must all reside on the same platform. For example, the core group cannot contain a mixture of z/OS and distributed processes.

  If the core group contains non-z/OS processes, or if the core group is composed of members that are at different version levels of the product, then you cannot use XCF for this core group.
- If the core group needs to be bridged to another core group, using the core group bridge service, then all of the core groups that are bridged to this core group are also homogeneous core groups.
- All members of the core group must be at Version 7.x of the product. If any members of the core group are running at a Version 6.x level of the product, then you must update them to Version 7.x, before you can switch to the alternative protocol provider.

Core group protocol versions

Core group members interact with each other through a variety of protocols such as the discovery protocol, the failure detection protocol, and the view synchrony protocol. Each of these protocols define a set of formatted messages that core group members exchange according to a common algorithm.

Note: This topic references one or more of the application server log files. Beginning in WebSphere Application Server Version 8.0 you can configure the server to use the High Performance Extensible Logging (HPEL) log and trace infrastructure instead of using `SystemOut.log` , `SystemErr.log`, `trace.log`, and `activity.log` files or native z/OS logging facilities. If you are using HPEL, you can access all of your log and trace information using the LogViewer command-line tool from your server profile bin directory. See the information about using HPEL to troubleshoot applications for more information on using HPEL.

New protocol versions are added to the product if new messages, or new algorithms are required to support new product features, or to improve core group performance. Because the new messages, or new algorithm might not be compatible with the existing messages or algorithm, a new protocol might not be able to interoperate with the old version of the protocol.

There are two major categories or groups of protocols.

- A collection of lower level protocols, which are also referred to as the lower level wire format protocols. The setting for the IBM_CS_WIRE_FORMAT_VERSION core group custom property determines which protocol version is used for this group of protocols.

  Whenever the value specified for this property changes, a DCSV0005I message, similar to the following example, is sent to the SystemOut.log file, or, for the z/OS platform, to either SYSOUT or SYSPRINT:

  ```
  DCSV0005I: DCS Stack DefaultCoreGroup at Member
     DIAMONDCell01\DIAMONDCellManager01\dmgr: Started.
     Stack version information: DCSBV_WAS6_1_20060409.
     Stack protocol information: 61002.
  ```

  The Stack protocol information value at the end of this message indicates the protocol version that is being used. For example, a value of 51001 means the 6.0.0 version is being used. A value of 61002 means the 6.1.0 version is being used.

  transition: In Version 7.0 of the product, HMGR0226I messages were used to indicate the wire format protocol version that was set for a core group. Starting with Version 7.0.0.1, HMGR0226I messages are used to indicate the high availability manager format protocol version that is set for a core group, and DCSV0005I messages are used to indicate the wire format protocol version that is set for a core group.

- A collection of higher level protocols, which are also referred to as the high availability manager protocols. The setting for the IBM_CS_HAM_PROTOCOL_VERSION core group custom property determines which protocol version is used for this group of protocols.

  Whenever the value specified for this property changes, a HMGR0226I message, similar to the following message, is sent to the SystemOut.log file, or, for the z/OS platform, to either SYSOUT or SYSPRINT:

  ```
  HMGR0226I: The core stack configuration parameter IBM_CS_HAM_PROTOCOL_VERSION
     has been set to 6.0.2.31.
  ```

  This message indicates that the high availability manager protocol version 6.0.2.31 is being used.

The protocol version settings for each of these two categories are independent of each other.

Whenever a new protocol version is added to the product:

- The new protocol version is assigned a unique ID.
- Support for the existing, in-service protocol versions is retained because all of the members of a core group might not be running at the same product level.
- The oldest supported protocol version is the default protocol version. Older protocol versions are deprecated using the standard product deprecation rules.
- You must explicitly update either the IBM_CS_WIRE_FORMAT_VERSION or the IBM_CS_HAM_PROTOCOL_VERSION core group custom property to change the protocol version that is used with a core group.

  gotcha: You cannot update to a newer protocol version until all of the members of this core group are upgraded to a code level that supports the newer protocol version.

When to select a new core group protocol version

Core group protocol versions are always cumulative. Any functional enhancement that is provided in a previous protocol, is included in any subsequent protocols. It is recommended that you always use the latest protocol version whenever possible. But, before configuring the members of a core group to use a new protocol version, you must make sure that all of the core group members are running at a WebSphere Application Server code level (VRM) that is equal to or greater than the new protocol version.

When you have determined that you can use a newer core group protocol level with a particular core group, use either the IBM_CS_WIRE_FORMAT_VERSION or the IBM_CS_HAM_PROTOCOL_VERSION core group custom property to configure all of the core group members to run with that newer version. You can change the value of this property while the core group members continue to run. After you save and synchronize the changed value to all of the nodes containing core group processes, the high availability manager automatically detects the configuration change and starts using the new core group protocol version to communicate with the core group members.

gotcha: You do not have to restart the core group members when you move from an older protocol version to a newer protocol version. However, you must restart all of the core group members if you move from a newer protocol version to an older protocol version.

Using the high availability manager protocol to establish transparent bridge failover support

Core group bridges provide the mechanism that is used to represent and manage the cross core group state that is used by WebSphere Application Server components. Part of the management process for this cross-core group state is to perform core group bridge state rebuilds whenever there is a change in the number of running core group bridges in a topology. The core group bridge state rebuild is the means by which core group bridges calculate the ownership, and distribution of the cross-core group state among the running set of bridges.

During core group bridge state rebuilds, cross-core group state can be moved between running bridges. This situation might cause the data to be temporarily unavailable until the bridge has completed the rebuild process. Common symptoms of this problem include:

- JNDI lookups failing.
- A WebSphere proxy server, or on demand router generates a 503 response code after a core group bridge failover occurs
- The following array index out-of-bounds exception occurs:

```
[7/9/08 17:12:20:749 EDT] 00000030 UserCallbacks E
HMGR0142E: An error occurred in a component called back by the High Availability Manager
The exception is java.lang.ArrayIndexOutOfBoundsException at
com.ibm.ws.cluster.propagation.bulletinboard.BBDescriptionManager.getOrderedBytes(BBDescriptionManager.java:618)
```

bprac: If you are running on Version 7.0.0.1 or later, set the IBM_CS_HAM_PROTOCOL_VERSION core group custom property to `6.0.2.31` for all of your core groups to avoid a possible high availability state outage during core group bridge failover. When this custom property is set to `6.0.2.31`, the remaining bridges recover the high availability state of the failed bridge without the data being unavailable in the local core group.

gotcha:

- Ensure that all core groups that are connected with core group bridges are running the same protocol version.
- Transparent bridge failover is designed to hold state data constant during core group bridge rebuilds along the state data path, which is the path that consists of the state provider, one core group bridge in each respective core group, and a state data consumer. Failure scenarios involving core groups with no remaining active bridges might still result in temporary state outages.

Determining which protocol version to use

bprac: It is recommended that you always use the newest protocol version whenever possible. This practice is especially critical for large topologies because most of the recent protocol changes include scalability improvements. However, before configuring the members of a core group to use a new protocol version, you must verify that all of the core group members are running at a product code level (VRM) that is equal to or greater than the VRM in which the desired protocol version was added to the product. For example:

- A core group that only contains Version 7.0 core group members, or contains a mixture of Version 6.1.0.0 and 7.0 core group members can be configured to use either the Version 6.0.0, 6.0.2.9, or 6.1.0 wire format protocol.
- A core group that contains a mix of Version 6.1.0.19 and 7.0.0.1 core group members can be configured to use the Version 6.0.2.31 high availability manager protocol.

Supported core group protocol version IDs

The following tables summarize for each protocol category the minimum level of the product that core group members must be running at before they can be associated with a specific protocol version. These tables also describe the new capabilities that were added in each protocol version.

Use these tables to determine which protocol versions you can use with a particular core group, and then use either the IBM_CS_WIRE_FORMAT_VERSION or the IBM_CS_HAM_PROTOCOL_VERSION core group custom property to configure all of the members of that core group to run using the newest of those protocol versions that is supported by the level of the product on which you are running. The high availability manager automatically detects the configuration changes and starts to use the new core group protocol version with these core group members.

depfeat: Wire format protocol versions 6.0.0 and 6.0.2.9 are deprecated. Whenever possible you should use a newer protocol version.

Table 1. Supported wire format protocol version IDs. The protocol version ID indicates the first version, release, and modification level in which that version is included. The following table lists the supported wire format protocol version IDs.

Version ID	Required Minimum product Level	Description
6.0.0	Any	This protocol version is the original or base version. All versions of the high availability manager can use this protocol. If you do not specify a particular wire format protocol version, the high availability manager uses this version.
6.0.2.9	6.0.2.9 for Version 6.0.2, 6.1.0.0 for Version 6.1, 7.0.0.0 for Version 7.0.	This protocol version was included in the 6.0.2.9 service pack, and in Versions 6.1, and 7.0 of the product to facilitate core group bridge scalability. This version is recommended for large topologies that contain multiple core groups and core group bridges as part of their configuration.
6.1.0	6.1.0.0 for Version 6.1, 7.0.0.0 for Version 7.0	This protocol version was included in Versions 6.1, and 7.0 of the product. This version adds core group scalability improvements, and more support for large topologies.

Table 2. Supported high availability manager protocol version IDs. The protocol version ID indicates the first the version, release, and modification level in which that version is included. The following table lists the supported high availability manager protocol version IDs.

Version ID	Required Minimum product Level	Description
6.0.2.31	6.0.2.31 for Version 6.0.2, 6.1.0.19 for Version 6.1, 7.0.0.1 for Version 7.0	This protocol version is the original or base version of the high availability manager protocol, and is available in the 6.0.2.31, 6.1.0.19, and 7.0.0.1 and higher versions of the product to facilitate core group bridge scalability. This protocol version is recommended for topologies that contain multiple core groups and core group bridges as part of their configuration.

High availability groups

High availability groups are part of the high availability manager framework. A high availability group provides the mechanism for building a highly available component and enables the component to run in one of several different processes. A high availability group cannot extend beyond the boundaries of a core group.

A high availability group is associated with a specific component. The members of the group are the set of processes where it is possible to run that component. Therefore, a product administrator cannot directly configure or define a high availability group, and its associated set of members. Instead high availability groups are created dynamically at the request of the components that need to provide a highly available function.

Scope

A high availability group cannot extend beyond the boundaries of a core group. Therefore, a highly available component cannot fail over from a server process that is defined in one core group to a server process that is defined in a different core group.

Life cycle

Because high availability groups are dynamically created, a product administrator has no direct control over when they are created or destroyed. A high availability group is created when component code that runs in a given process calls the high availability manager framework to join a group. The calling component must provide the name of the high availability group for the high availability manager framework to join.

If a high availability group with this name does not currently exist, the high availability manager creates one, and makes this member the first member of the newly created group. If the high availability group already exists, this member is added to the set of high availability group members.

Because several different components might use the high availability manager framework, it is possible to have several different high availability groups across the same set of processes. However, each high availability group always has a unique group name.

A high availability group ceases to exist when all of the group members leave the group, which typically occurs when all of the processes that host members of a given high availability group stop.

Group name

Every high availability group has a unique name. Because any component can create a high availability group for that component to use, it is the high availability group name that ties a given component to a particular high availability group. A high availability group name is not a simple string; this name is a set of name-value pairs that the creating component specifies. A high availability group name can look like the following example:

```
Company=IBM,ComponentName=TM,policy=DefaultNoQuorumOneOfNPolicy
```

A component can specify any number of name-value pairs to create a unique name for their high availability group.

Member state

Each member of a high availability group is either idle, active or disabled. Typically, a high availability group member will be either idle or active. A member that is idle is not assigned any work, but is available as a backup if a member that is active fails. A member that is active is designated as the member to handle the component workload.

If a member is disabled, it cannot participate in the high availability group. A disabled member is not assigned any work, and is not available as a backup if an active member fails. An administrator might disable a member if they plan to remove, delete, or cycle power on the associated server. However, this action is not required.

Policy

Every high availability group has an associated policy. The policy is used to determine which members of a high availability group are active at a given point in time. The policies available for high availability groups to use are stored as part of the core group configuration.

High availability group policies

Every high availability group has a policy associated with it. This policy is used to determine which members of a high availability group are active at a given time.

The policies that the high availability groups use are stored as part of the core group configuration. The same policy can be used by several different high availability groups, but all of the high availability groups to which it applies must be part of the same core group. Before modifying or deleting an IBM provided policy, read the topic *High availability group policy modification guidelines*.

Policy selection

Policies are statically configured, and the high availability groups that are governed by them are created dynamically. Therefore, a mechanism is required to associate a running high availability group to a configured policy. This association is accomplished by comparing the following two pieces of information.

- The high availability group name
- The policy match criteria

The *High availability group policy modification guidelines* topic provides more detail about how a policy is selected.

Policy settings

Some policy settings apply for all policy types while others only apply for specific policy types. Some of the policy settings also influence the overall behavior of a policy. The topic *Implications of high availability* *group policy settings* describes the various settings, the applicable policy types, and how they influence policy behavior.

Policy enforcement

Whenever one of the following conditions occurs, the high availability manager runs the policy that is associated with a high availability group and takes any appropriate action:

- A member joins or leaves that high availability group. A member leaves the group if the member fails.
- The state of a member of that high availability group changes. For example, if the state changes from idle to active, or from idle to disabled, the policy rules are reapplied.

Policy changes

The high availability manager dynamically detects policy configuration changes. Therefore, policy setting changes go into affect as soon as you save and propagate these changes. Server restarts are not required.

High availability group policy selection process

Every high availability group has a unique group name that consists of a set of name-value pairs. Every policy definition contains an attribute called *match criteria* that is also a set of name-value pairs. To determine the policy for a high availability group, the group name is compared to the match criteria of all the associated core group polices. The policy with the strongest match to the group name is assigned to the high availability group:

Note: This topic references one or more of the application server log files. Beginning in WebSphere Application Server Version 8.0 you can configure the server to use the High Performance Extensible Logging (HPEL) log and trace infrastructure instead of using `SystemOut.log` , `SystemErr.log`, `trace.log`, and `activity.log` files or native z/OS logging facilities. If you are using HPEL, you can access all of your log and trace information using the LogViewer command-line tool from your server profile bin directory. See the information about using HPEL to troubleshoot applications for more information on using HPEL.

When selecting a policy for a high availability group, the high availability manager:

1. Finds the set of policies that are eligible to govern a high availability group. For a policy to be eligible, all name-value pairs in the match criteria of an eligible policy must be contained in the name of the high availability group.
2. Selects the policy that has the most name-value pair matches from the list of eligible policies, and uses that policy to govern the high availability group.

Any component can create a high availability group for that component to use. However, the component code must specify the name-value pairs that are used for the high availability group name. The product administrator can control the name-value pairs that make up a policy match criteria, and thereby control which policy governs a particular high availability group.

The product includes a couple of predefined policies. The following examples demonstrate the matching mechanism that is used for these policies.

Clustered TM Policy

The transaction manager component uses the policy Clustered TM Policy when the component is configured for high availability. The following description illustrates why, under these conditions, this policy is selected for the transaction manager high availability group:

- A cluster member process, such as ServerA, is started.

- The transaction manager component code joins a high availability manager to the high availability group named:

  ```
  GN_PS=testCell\testNode\ServerA,IBM_hc=MyCluster,type=WAS_TRANSACTIONS
  ```

- ServerA is defined as a member of the DefaultCoreGroup core group for which the following policies are defined:
  - Clustered TM Policy, which has the match criteria type=WAS_TRANSACTIONS.
  - Default SIBus Policy, which has the match criteria type=WSAF_SIB.
- The high availability manager compares the group name to the match criteria for the two available policies. The high availability manager eliminates the Default SIBus Policy because the match criteria is not a proper subset of the high availability group name. The high availability manager determines that Clustered TM Policy is the closest match because:
  1. The match criteria for that policy includes the name-value pair type=WAS_TRANSACTIONS, which is also specified in the high availability group name. Therefore, the match criteria is a proper subset of the high availability group name.
  2. The match criteria for that policy more matches (one) than the match criteria for Default SIBus Policy, which is eliminated because it does not have any matches.

Administrator TM Policy

This example builds on the previous example and demonstrates how an administrator can define a new policy to govern the transaction manager high availability group. In this example the same high availability group name and default policies that are described in the previous example are used. However, in this example, the administrator creates a new policy in the DefaultCoreGroup configuration called the Administrator TM Policy. For the high availability manager to select this new policy, the policy must be eligible and contain more matches than any other policy.

The following description illustrates why, under these conditions, the policy Administrator TM Policy is selected for the transaction manager high availability group:

- The cluster member process ServerA is started.
- The transaction manager component code joins a high availability manager to the high availability group named:

  ```
  GN_PS=testCell\testNode\ServerA,IBM_hc=MyCluster,type=WAS_TRANSACTIONS
  ```

- ServerA is defined as a member of the DefaultCoreGroup core group, for which the following policies are defined:
  - Clustered TM Policy, which has the match criteria type=WAS_TRANSACTIONS.
  - Default SIBus Policy, which has the match criteria type=WSAF_SIB.
  - Administrator TM Policy, which has the match criteria IBM_hc=MyCluster,type=WAS_TRANSACTIONS.
- The high availability manager compares the group name to the match criteria for the available policies. The high availability manager eliminates the Default SIBus Policy because the match criteria is not a proper subset of the high availability group name. It determines that Clustered TM Policy and Administrator TM Policy are both eligible policies, because their match criteria are proper subsets of the high availability group name:
  - Clustered TM Policy contains the name-value pair type=WAS_TRANSACTIONS, which is also specified in the high availability group name.
  - Administrator TM Policy contains the name-value pairs IBM_hc=MyCluster and type=WAS_TRANSACTIONS, which are both specified in the high availability group name.

  Because Administrator TM Policy has two matching pairs, IBM_hc=MyCluster and type=WAS_TRANSACTIONS, and Clustered TM Policy has only one matching pair, type=WAS_TRANSACTIONS, the high availability manager associates Administrator TM Policy with the transaction manager high availability group.

Ambiguous Matches

Do not configure identical match criteria for multiple policies in the same core group. Configuring identical match criteria causes an ambiguous match to the associated high availability group. Because a high availability group can only be associated with one policy, if the previously described matching mechanism does not result in a single policy match, the high availability manager puts the high availability group in error state, and does not make any of the group members active. Depending on the nature of the problem, the high availability manager might write one of the following error messages to the `SystemOut.log` file:

```
HMGR0301W: No policy was located for the group named {0}
HMGR0302W: Multiple policies match the group named {0}, Matching Policies are {1}
```

You can use the administrative console to view the policies associated with a high availability group and the current state of members of that group.

Implications of high availability group policy settings

All of the settings that are specified for a policy affect how the high availability manager governs a high availability group associated with that policy. Some of the policy settings are policy type specific, while others apply to all policy types. It is important to understand the implications for all of the associated high availability groups before you change the settings of an existing policy.

Implications of the Policy type setting

The policy type determines which members of a high availability group are automatically made active when the servers containing these members start. You can not directly change the policy type of an existing high availability group policy. If you need to change the policy type, you must create a new policy with a different policy type and give it a match criteria that makes the high availability manager select a new policy instead of the original one to associate with the high availability group.

Before creating a new policy with a different policy type, you must determine which components are using the high availability groups that are governed by the original policy and make sure that those components support the new policy type. For example, the service integration bus (SIB) component might require a One of N policy for its high availability group, because it only wants one group member active at a given time. If you change the policy that is associated with the service integration bus high availability group to be an All Active policy, the service integration bus high availability support might not function properly and data corruption can occur.

You can select one of the following policy types when you create a new policy:

All active policy
When this policy is selected, all of the members of the high availability group are made active.

M of N policy
When this policy is selected for a high availability group with N members, M of them are made active. The number that M represents is configurable in the policy settings. You can use the Preferred servers setting to designate the preference order in which members of the high availability group are made active.

No operation policy
When this policy is selected, none of the high availability group members are made active. You can use the administrative console to manually activate specific group members.

One of N policy
When this policy is selected for a high availability group with N members, only one member of the group is made active. You can use the Preferred servers setting to designate the preference order in which members of the high availability group are made active.

Static policy

When this policy is selected, only the members specified in the Static group servers setting are made active.

gotcha: Only the Static, One of N, and No operation policies apply for service integration and messaging engines. See the information about policies for service integration.

Implications of the Preferred servers setting

With the One of N and M of N policy types, you can set up a list of preferred servers as part of the policy settings. The preferred server list enables an administrator to indicate a preference as to which high availability group member is made active. If no preferred server list is specified, any of the available high availability group members can be selected as the member to activate. If a preferred server list is specified, then the member to activate is selected from this list, in order of preference. The most preferred server is the first one on the list. The following example demonstrates how a policy uses of the preferred server list.

Example:

A high availability group has three members that are located on application servers named ServerA, ServerB, and ServerC. This group is governed by a One of N policy, under which only one of the three members can be active at a given time. When all three members are running and available at the time that the policy is enforced:

- If no preferred servers are specified, the high availability manager randomly selects one of the three members and makes it active.
- If ServerB is the only server on the preferred servers list, the high availability manager makes the member located on this server active before either of the other two members, provided the member located on this server is available when the policy is enforced.
- If all three of the application servers are listed on the preferred servers list in the following order, and if all other things are equal, the high availability manager makes the member that is located on ServerC active:

  ServerC

  ServerA

  ServerB

The two other policy settings that directly affect how the preferred server list is used are the Failback and Preferred servers only settings.

Implications of the Failback setting

The Failback setting is used to specify what happens to the high availability group member on the most preferred server when it is restarted following a failure. The affect of the Failback setting on a member is best demonstrated with two examples.

During startup example:

A high availability group has three members that are located on application servers named ServerA, ServerB, and ServerC. This group is governed by a One of N policy, under which only one of the three members can be active at a given time. The server named ServerB is the only server on the preferred server list. In this example, none of the servers are started.

When ServerA starts, the One of N policy dictates that the high availability manager makes a member active. Because this application server is the only server running, the member on ServerA is activated. When we start ServerB, which is the only server on the preferred server list, one of two things happens:

- If Failback is enabled when ServerB starts, the high availability manager deactivates the currently active member and activates the member on ServerB, because ServerB is on the preferred server list.
- If Failback is disabled when ServerB starts, the currently active member remains the active member.

Following a failure example:

A high availability group has three members that are located on application servers named ServerA, ServerB, and ServerC. This group is governed by a One of N policy, under which only one of the three members can be active at a given time. ServerB is the only server on the preferred server list and is the only member that is currently active. If ServerB fails, the high availability manager activates one of the remaining members to replace that member. The Failback setting determines what happens after ServerB is repaired and restarted.

- If Failback is enabled when ServerB restarts, the currently active member is deactivated, and the member on ServerB is activated, because ServerB is still the most preferred server
- If Failback disabled when ServerB restarts, the currently active member remains the active member.

Implications of the Preferred servers only setting

The Preferred Servers Only setting is used to instruct the policy to activate members on preferred servers only. With this setting enabled, only members running on the servers that are specified in the preferred servers list are activated. If no preferred servers are specified, or no preferred servers are currently available, then no members are activated.

During startup example:

A high availability group has three members that are located on application servers named ServerA, ServerB, and ServerC. This group is governed by a One of N policy, under which only one of the three members can be active at a given time. ServerB the only server on the preferred server list. In this example, none of the servers are started.

When ServerA starts, the One of N policy dictates that the high availability manager activates a member. Because ServerA is the only server running, the member on ServerA is activated. Because it is the only server on the preferred server list, when ServerB starts, one of two things happens:

- If Preferred servers only is enabled when ServerA or ServerC starts, no member is activated because the high availability manager can only activate a member that is located on a server that is on the preferred servers list. When ServerB starts, the high availability manager activates the member on ServerB because ServerB is on the preferred servers list.
- If Preferred servers only is disabled when ServerA starts, the member on ServerA is activated because any member of the group can be the active member. When ServerB or ServerC starts, no activation occurs because the member on ServerA is already active.

Following a failure example:

A high availability group has three members that are located on application servers named ServerA, ServerB, and ServerC. This group is governed by a One of N policy, under which only one of the three members can be active at a given time. ServerB the only server on the preferred server list. The member located on ServerB is the active member. If ServerB fails, one of two things happens:

- If Preferred servers only is enabled when ServerB fails, the high availability manager can only activate another members that is located on a server that is included on the preferred servers list. Because ServerB is the only server on the preferred servers list, no other member is activated.
- If Preferred servers only is disabled when ServerB fails, the high availability manager activates one of the remaining members to replace the member on ServerB.

Implications of the Static group servers setting

You can specify a list of static group servers as part of the configuration settings for a static policy type. When a high availability group is governed by a static policy type, the static group server list defines which group members are activated if it is possible to do so.

Implications of the Is alive timer setting

The Is alive timer setting controls how frequently the high availability manager checks the health of the active group members that are governed by a given policy. The high availability manager can detect two fundamentally different kinds of failures:

- It can detect when an entire process stops functioning or terminates. This type of failure detection does not depend on the value that is specified for the Is alive timer setting.
- It can detect when a program fails. This type of failure detection does depend on the value that is specified for the Is alive timer setting. The value that is specified for the Is alive time setting determines the amount of time that might pass before a processing problem that does not cause the entire process to stop functioning or terminate is detected.

The administrator has the ability to specify the Is alive timer at the policy level, where it applies to all the members that are governed by this policy, at the process level where it applies to all members running in a particular process. The administrator can also disable this type of failure detection at either of these levels.

Implications of the Quorum setting

Quorum is a mechanism that you can use to protect resources that, in the event of a failure, are shared across members of a high availability group. When enabled, the policy does not activate any group members until quorum is achieved. A high availability group does not achieve quorum until a majority of the members are running. For example, if there are n members in a group, (n/2) + 1 servers must be online to achieve quorum.

Quorum is an advanced function that is designed to work with clusters, specialized component code, and a hardware control facility. Currently, none of the high availability groups supporting product components use the quorum mechanism. Therefore, do not enabled the Quorum setting.

High availability group policy modification guidelines

Use the following guidelines to help determine when to create a new high availability group policy, and when to modify or delete an existing policy.

Do not delete the default IBM provided policies

If you want to override one of the default policies that IBM provides, it is recommended that you do not delete the current policy. Instead, create a new policy with more specific match criteria. The policy with the greatest number of matches is the one that is used. Not deleting the IBM provided policy enables you to revert back to that policy if a problem occurs with your new policy.

Do not try to change the type of an existing policy

After a policy for a high availability group is created, you can change some of the policy attributes such as preferred servers, or failback, but you cannot change the policy type. If you need to change the policy type, you must create a new policy and then use match criteria to associate it with the appropriate high availability group. The *High availability group policy selection process* topic describes how the high availability manager selects a policy for a high availability group.

Make sure that you know which policies a component using that high availability group supports before creating a new policy to associate with that group

A component does not necessarily support all policy types and options. Therefore, before changing the policy that is associated with a given high availability group, make sure you fully understand if the application server code using that high availability group supports the change. For example, if you want to change the type of policy that is associated with the high availability group used by the transaction manager component, make sure the transaction manager code supports the new policy type before making the change.

Do not use the same match criteria for multiple policies in the same core group

If you have multiple policies configured with identical match criteria, the policy match to the associated high availability group is ambiguous. If you are creating a new policy to replace an older policy that you created, you might need to delete the older policy to specify the appropriate match criteria. Another alternative is to specify additional match criteria in each policy so no ambiguity exists as to which policy is controlling the high availability group.

Chapter 4. Disabling or enabling a high availability manager

A unique HAManagerService configuration object exists for every core group member. The enable attribute in this configuration object determines if the high availability manager is enabled or disabled for the corresponding process. When the enable attribute is set to `true`, the high availability manager is enabled. When the enable attribute is set to `false`, the high availability manager is disabled. By default, the high availability manager is enabled. If the setting for the enable attribute is changed, the corresponding process must be restarted before the change goes into effect. You must use the wsadmin tool to disable or enable a high availability manager.

Before you begin

Determine if you need to use a high availability manager to manage members of a core group.

Note: When you disable the high availability manager, make sure that you disable it for a core group so that all members are disabled. Do not disable the high availability manager for select members of the core group. For more information, see the following documents:

- "2.1.6.1 High availability is not needed" section of the "Best Practices for Large WebSphere Topologies" document
- "6.3.3 Partitioned cell with HA manager-disabled core group topology" section of the "Techniques for Managing Large WebSphere Installations" Redbook

About this task

You might want to disable a high availability manager if you are trying to reduce the amount of resources, such as CPU and memory, that the product uses and have determined that the high availability manager is not required on some or all of the processes in a core group.

You might need to enable a high availability manager that you previously disabled because you are installing applications on core group members that must be highly available.

Complete the following steps if you need to disable a high availability manager or to enable a high availability manager that you previously disabled.

Procedure

1. In the administrative console, navigate to the Core group service page for the process.
   - For a deployment manager, click **System Administration > Deployment manager > Core group service**.
   - For a node agent, click **System Administration > Node agent >** *node_agent* **> Core group service**.
   - For an application server, click **Servers > Server Types > WebSphere application servers >** *server_name* **> Core group service**.
2. If you want to disable the high availability manager for this process, deselect the `Enable service at server startup` option.
3. If you want to enable the high availability manager for this process, select the `Enable service at server startup` option.
4. Click **OK** and then click **Review**.
5. Select **Synchronize changes with nodes**, and then click **Save**.
6. Restart all of the processes for which you changed the Enable service at server startup property setting.

Results

The processes start with the high availability manager in the changed state.

What to do next

To verify that the high availability manager is in the proper state, check the log file for one of the following messages:

```
HMGR0005I: The Single Server DCS Core Stack transport has been started for core group DefaultCoreGroup.
```

This message indicates that the high availability manager is disabled because the high availability manager communication transport can only establish communication with a single server process.

```
HMGR0001I: The DCS Core Stack transport has been started for core group DefaultCoreGroup. There are x members.
```

This message indicates that the high availability manager is enabled because the high availability manager communication transport can establish communication with multiple server processes. *x* indicates the number of server processes with which communication is established.

Chapter 5. Viewing high availability group information

High availability groups are dynamically created components of a core group. They cannot be configured directly, but they are directly affected by static data, such as policy configurations, which is specified at the core group level. You can use the administrative console to view information about the high availability groups that are part of a core group.

Before you begin

You need to know the name of the core group that contains the high availability groups you want to view. You should also determine if you need to view all of the high availability groups or a subset of these groups, based on the group name.

About this task

You might want to perform this task if:

- You want to view the current set of high availability groups.
- You want to view the group name of a high availability group.
- You want to view the policy that is associated with a high availability group.
- You want to view the state of a high availability group

To view information about the high availability groups contained in a core group:

Procedure

1. In the administrative console, click **Servers > Core Groups > Core group settings**.
2. Click the core group that contains the high availability groups you want to view.
3. Click the **Runtime** tab.
4. Specify a value in the **Group name** field.
   - Specify an asterisk in this field if you want a list of all the high availability groups that are part of this core group.
   - Specify a set of name-value pairs, separated by a comma, to get a list of only those high availability groups that contain the specified name-value pairs in their group name. For example, you might specify the following value to obtain a list of all of the high availability groups that contain `IBM_hc=MyCluster` and `type=WAS_TRANSACTIONS` in their names.

     ```
     IBM_hc=MyCluster,type=WAS_TRANSACTIONS
     ```
5. Click **Show groups**.

Results

A list of high availability groups that are contained in this core group that meet the specified criteria is displayed, with pertinent information about these groups.

What to do next

You can click on the name of one of the high availability group names to display information about members of that group. Because high availability group members are dynamically created, no member information is available for configured servers that are not actually running.

You can also view the distribution of active high availability group members.

Viewing the distribution of active high availability group members

Because high availability group members are dynamically created, core group policy configuration is used to determine which high availability group members the high availability manager activates. In some situations, it is possible for active members of multiple high availability groups to all be running on the same server process. You can use the administrative console to view the distribution of active high availability group members across your application servers.

Before you begin

You need to know the name of the core group that contains the high availability groups you want to view. You should also determine if you need to view all of the high availability groups or a subset of these groups, based on the group name.

About this task

You might want to perform this task if:

- You want to view the current active high availability group member distribution for the servers in a core group.
- You want to determine whether a particular server is overloaded because it is hosting the active member for multiple high availability groups.
- You want to check the current active member distribution before you update policies that might affect this distribution.
- You want to verify that you obtained the proper results to the current active member distribution after a policy change goes into affect.

To view the distribution of active high availability group members:

Procedure

1. In the administrative console, click **Servers > Core Groups > Core group settings**.
2. Click the core group that contains the high availability groups you want to view.
3. Click the **Runtime** tab.
4. Specify a value in the **Group name** field.
   - Specify an asterisk in this field if you want a list of all the high availability groups that are part of this core group.
   - Specify a set of name-value pairs, separated by a comma, to get a list of only those high availability groups that contain the specified name-value pairs in their group name. For example, you might specify the following value to obtain a list of all of the high availability groups that contain `IBM_hc=MyCluster` and `type=WAS_TRANSACTIONS` in their names.

     ```
     IBM_hc=MyCluster,type=WAS_TRANSACTIONS
     ```
5. Click **Show servers**.

Results

A list of servers displays that shows the number of active high availability group members on each server.

Servers with active members collection

Use this page to determine how many high availability group members are active on a particular application server.

To view this administrative console page, click **Servers > Core Groups > Core group settings >** *core_group_name*. Click on the **Runtime** tab and specify group name properties for a high availability group. (You can specify an asterisk (*) to get a list of the servers that are hosting active members for all the high availability groups in this core group.) Then select **Show severs**.

Server

Specifies the name of a server on which there are active high availability group members. This field is read-only.

Node

Specifies the node on which each server is running. This field is read-only.

Version

Specifies the version of the product on which each node is running. This field is read-only.

Active members

Specifies the number of high availability group members that are currently active on that server. This field is read-only.

High availability groups collection

Use this page to view information about the high availability groups contained in a core group.

To view this administrative console page, click **Servers > Core Groups > Core group settings >** *core_group_name*. Click on the **Runtime** tab. In the Group name properties field, specify a match criterion for a specific high availability group, or specify an asterisk to get a list of all the high availability groups that are part of this core group. (A match criterion is one or more name=value pairs that match attributes contained in a high availability group's name.) Then click **Show groups**.

To manage one of the listed high availability groups, select the name of the high availability group and one of the following buttons.

Button	Resulting action
Disable	Disables all of the members of a high availability group that were previously active or idle. One of the few times you might want to use this button is if you are planning to remove or delete all of the servers on which this group has a member running.
Enable	Enables all of the members of a high availability group that were previously disabled. These members can then be activated according to the policy associated with that group.

High availability group

Specifies the names of the high availability groups. The name of a high availability group is a set of name-value pairs or attributes, separated with commas. For example, `name=productiongroup,policy=abc,ibm=websphere` could be the name of a high availability group. This field is read-only.

Quorum

Specifies if quorum is enabled for each high availability group. This field is read-only.

Policy

Specifies the policies that have match criteria that matches properties contained in the name of that high availability group. There should only be one policy listed for a high availability group. However, if multiple policies have match criteria that equally match properties in a high availability group's name, all of the policies with matching criteria are listed, and the ERROR icon appears in the status column.

For example, if you have a high availability group named `name=productiongroup,policy=abc,ibm=websphere`, and MyPolicy1 has the match criteria `name=productiongroup`, and MyPolicy2 has the match criteria `policy=abc`, both MyPolicy1 and MyPolicy2 are considered matching policies and are listed in the Policy column.

This field is read-only.

Status

Specifies, with icons, whether or not only one policy is associated with a high availability group. If the OK icon displays in this column, a single policy is associated with that high availability group. If the ERROR icon displays in this column, multiple policies are associated with that group.

If the ERROR icon displays for a high availability group, you must adjust the match criteria for one or more of the policies listed in the Policy column for that group so that the correct policy is the only one associated with that high availability group.

The match criteria for multiple policies can match some of the same properties in a group's name as long as one policy has a match criteria that matches more of the properties in that group's name than the match criteria of any of the other policies. For example, if you have a high availability group with a name that consists of the following name and value pairs:

```
name=productiongroup,policy=abc,ibm=websphere
```

and MyPolicy1 has the match criteria `name=productiongroup` and MyPolicy2 has the match criteria `name=productiongroup,ibm=websphere`, MyPolicy2 is considered the matching policy because it has more match criteria that matched the properties contained in the name of the high availability group.

This field is read-only.

High availability group members collection

Use this page to view information about the individual members of a high availability group. This page lists the current members of the selected high availability group.

To view this administrative console page, click **Servers** > **Core Groups** > **Core group settings** > *core_group_name*. Click on the **Runtime** tab. In the Group name properties field, specify a match criterion for a specific high availability group, or specify an asterisk (*) to get a list of all the high availability groups that are part of this core group. (A match criterion is one or more name-value pairs that match attributes contained in the name of a high availability group.) Then click **Show groups** and select one of the listed high availability groups.

Button	Resulting action
Disable	Prevents a group member from participation in the group. A member in the disabled state can never be made active or used by the group.
Enable	Enables a group member that was previously disabled.
Activate	Activates an idle group member. This selection is only valid if the member belongs to a high availability group that is governed by a No operation policy.
Deactivate	Makes an active group member idle. This selection is only valid if the member belongs to a high availability group that is governed by a No operation policy.

Name

Specifies the name of a high availability group member.

Node

Specifies the node on which each high availability group member is running.

Version

Specifies the version of the product on which each node is running.

Status

Specifies the state of the high availability group members.

High availability group members are either idle, activated, or disabled. The usual states are idle or activated. One of the few times you might want to disable a member is if it is running on a server that you plan to remove or delete.

- If a group member is idle, it cannot be assigned any work.
- If a group member is activated, it can be assigned work.
- If a group member is disabled, it must be enabled before it can be activated.

Chapter 6. Creating a policy for a high availability group

Every high availability group has to have an associated policy. This policy determines which members of a high availability group to put in the active state.

Before you begin

Before creating a new policy, you should review the following topics:

-
- "High availability group policy modification guidelines" on page 35
- "High availability group policies" on page 29

You should also know:

- The name of the core group that you want to associate with the new policy.
- The name of the high availability group that you want this policy to control.
- The function, such as transaction log recovery or messaging engine, that is associated with this high availability group.
- The policy types, such as One of N or Static, that this function supports.
- The type of policy you want to create.
- The policy settings, such as failback, and preferred servers only, that you want to configure for this policy.

About this task

The product includes default policies that are already associated with the high availability groups some of the product components use. If these default policies do not meet the requirements of your installation, it is recommended that you create a new policy instead of changing one of the default policies. The creation of new policies provides you with the capability to tailor the policy settings to your installations requirements while giving you the option to revert back to the default policy.

To create a new policy:

Procedure

1. In the administrative console, click **Servers > Core Groups > Core group settings >** *core_group_name* **> Policies > New**.
2. Select the new policy you want in effect for a specific high availability group. If you need to define a new policy, the policy options are:
   - All active policy: All of the group members are activated.
   - M of N policy: *M* group members are activated. The number that is represented by *M* is defined as part of the policy details.
   - No operation policy: No group members are activated.
   - One of N policy: Only one group member is activated.
   - Static policy: The active members of a group are statically configured.

   For messaging engines in service integration, only the Static, One of N, and No operation policies apply.
3. Click **Next**.
4. Specify a name for the policy in the Name field. The name must be unique within the scope of the core group. Make the name meaningful to other administrators.
5. Optional: Specify a Description of the policy in the description field. This description might include the name of the associated core group.

6. Specify a value for the Is alive timer field, if the default value is too long or too short a time period. This value determines how frequently the high availability manager checks the health of the high availability group members. The default value is `0` seconds.
   - If you specify `-1` (minus 1), the Is alive timer is disabled.
   - If you specify `0` (zero), the value that is specified for the Is alive timer at the core group services level is used for high availability groups that are associated with this policy.
   - If you specify an integer between 1 and 2147483647, inclusive, this value is used for the high availability groups that are associated with this policy.
7. Make sure the Quorum field is not selected. You should not enable Quorum unless you are explicitly instructed to do so in the documentation for some other product.
8. Select the **Failback** field if you want to have the high availability manager make the most preferred member the currently active member whenever this action is possible. This option is available only for M of N and One of N policies.
9. Select the **Preferred servers only** field if you want the high availability manager to only activate group members on servers that are contained in the Preferred servers list. This option is available only for M of N and One of N policies. If you select this option, you must configure a list of preferred servers. A description of how to set up this list is provided in a later optional step.
10. Specify the number of group members that you want active in the Number of active members field. This option is available only for an M of N policy.
11. Click **Apply** and then select **Match criteria**.
12. On the next panel, click **New** and then configure the match criterion for this policy.
    a. In the Name field, specify the name of one of the name-value pairs contained in the name of the high availability group that you want to associate with this policy.
    b. In the Value field, specify the value of the name-value pair you specified in the Name field.
    c. Optional: In the Description field, add a description of this match criterion . For example, you might specify `First attribute` to indicate that this name-value pair matches the first attribute contained in the group name.
    d. Click **OK**.
    e. Repeat these steps for each additional attribute you want to include as part of your match criterion.

    You should set the match criterion for a new policy to two or more of the name-value attributes that are contained in the name of the high availability group to ensure that this policy is used instead of one of the product default policies. Using this example, the following high availability group-to-policy association is established:
13. In the Additional Properties section, select the **Static group servers** field to configure the list of servers that you want activated. This option is available only for Static policies. Click **Add** to move core group servers into the list of Static group servers, and then Click **OK** after you complete the list.
14. Optional: Under Additional Properties, select **Preferred servers** and select the preferred servers for this policy. This option is available only if you selected the Preferred servers only field for M of N and One of N policies. If you do not set up this list, no group members are activated.

    Click **Add** to move core group servers into the list of preferred servers.

    Select specific servers in the list and click **Move up** and **Move down** to adjust the order of the servers within the list. Make sure that the most preferred server is at the beginning of the list and the least preferred server is at the end of the list.

    After you complete the preferred servers list, click **OK**.

    Important: Use caution when selecting preferred servers. The product cannot detect if you select an inappropriate server as a preferred server. For example, if the policy affects a messaging engine or transaction service, only select preferred servers from the messaging engine cluster. Similarly, if the policy affects a transaction service, only select preferred servers from the transaction service cluster.

15. Click **OK** and then click**Review**.
16. Select **Synchronize changes with nodes**, and then click **Save**.

Results

The new policy goes into affect after it is saved and synchronized. You do not have to stop and restart the affected application servers.

What to do next

You can change the **Failback** and **Preferred servers only** options for this policy without stopping and restarting the affected application servers.

You can create or update the list of preferred servers that for this policy without stopping and restarting the affected application servers.

Core group policies

Use this page to create or update the various high availability group policies. For a given high availability group, the associated policy determines which members of the group should be made active.

To view this administrative console page, click **Servers** > **Core Groups** > **Core group settings** > **New**, or select an existing core group, and then click **Policies**.

Click **New** to define a new policy. After a policy is defined there are several fields that you can no longer change. To change those fields, delete and redefine the policy. Click **Delete** after selecting a policy to delete the selected policy.

After adding a high availability group, you need to take more actions to enable workload balancing for messaging resources. For more information about the extra actions, see the Related tasks.

All of the policy fields on this page are read-only. To change the values specified in any of these fields, click on the name of the policy you want to change. When the console page **Core group settings >** *core_group_name*> **Policies** > *policy_name* displays, and you can edit the policy properties.

Name

Specifies the name of the policy.

Description

Specifies a description of the policy.

Policy type

Specifies the desired policy type.

Restrictions:

1. If you are setting up a policy for a transaction manager, you must select One of N as the policy type.
2. If you are setting up a policy for service integration, you must select One of N, Static, or No operation as the policy type. The default service integration policy uses a One of N policy type.

Following is a list of valid policy types:

All active
The All active policy indicates that the high availability manager keeps all of the application components that are running on all of the servers in the high availability group active at all times.

M of N
The M of N policy is similar to the One of N policy. However, it enables you to specify the number (M) of high availability group members that you want to keep active if it is possible to do so. The number of active members must be greater than one and less than or equal to the number of servers in the high availability group. If the number of active servers is set to one, this policy is a match for the One of N policy.

No operation
The No operation policy indicates that no high availability group members are made active.

One of N
The One of N policy keeps one member of the high availability group active at all times. This is used by groups that desire singleton failover. If a failure occurs, the high availability manager starts the singleton on another server.

Static
The Static policy allows you to statically define or configure the active members of the high availability group.

Match criteria

Specifies one or more name-value pairs that are used to associate this policy with a high availability group. These pairs must match attributes that are contained in the name of a high availability group before this policy is associated with that group.

Core group policy settings

Use this page to define a policy for a high availability group. A policy is defined at the core group level. It only applies to matching high availability groups contained within this core group

To view this administrative console page, click **Servers** > **Core Groups** > **Core group settings** > **New** >, or select an existing core group, and then click **Policies** > *pollicy_name*.

Name

Specifies the name of the policy. This name must be unique within the scope of a core group.

Policy type

Specifies the policy type that was selected when this policy was created. This is a read-only field. If you want to change the policy type, you must delete this policy and then create it again specifying a different policy type. If this is an IBM provided policy, do not delete it. Instead create a new policy and specify more of the attributes contained in the name of the high availability group as the match criterion for this new policy. The policy with the greatest number of matches to attributes in a group's name is the policy that is associated with that group.

Description

Specifies a description of this policy. For example, the clustered TM policy provided with the product has "TM One-Of-N Policy" as its description.

Is alive timer

Specifies, in seconds, the interval of time at which the high availability manager will check the health of the active group members that are governed by this policy. If a group member has failed, the server on which the group member resides is restarted.

The high availability manager detects two fundamentally different kinds of failures.

- An entire process failure. This failure detection is accomplished using functions such as the heartbeat timers. This type of detection does not involve the Is alive timer function. If an entire process fails, it will be detected and the various high availability groups will fail over to other servers in the core group.
- An application or program failure. If, for some reason. an application or a program, like the transaction manager or a service integration bus function hangs, the high availability manager will eventually detect the hang. The amount of time that might pass before the hang is detected is determined by the value specified for the Is alive timer parameter. The parameter controls how often the high availability manager will call back to the component that created the high availability group member and ask if it is still alive. This allows detection of hung code or program errors that somehow do not cause the entire process to stop functioning or terminate.

Data type	Integer • Valid values are -1 to 600 seconds, inclusive. • If -1 (negative 1) is specified, this function is disabled. • If 0 (zero) is specified, the frequency at which the high availability manager checks the health of the active group members is determined by the time interval specified at the application server process level. • If a value larger than 0 (zero) is specified, the high availability manager uses the time interval specified here, instead of the one specified at the application server process level, when determining how frequently it should check the health of the high availability group members using this policy.
Default	0 (zero)

Fail back

Specifies whether work items assigned to the failing server are moved to the server that is designated as the most preferred server for the group if a failure occurs. This field only applies for M of N and One of N policies.

Preferred servers only

Specifies whether group members are only activated on servers that are on the list of preferred servers for this group. This field only applies for M of N and One of N policies.

Number of active members

Specify how many of the high availability group members are to be activated. This field only applies for the M of N policy.

Quorum

Specifies whether quorum checking is enabled for a group governed by this policy. Quorum is a mechanism that can be used to protect resources that are shared across members of the group in the event of a failure.

Important: Quorum is an advanced hardware function and should not be enabled unless you thoroughly understand how to properly use this function. If not used properly, this function can cause data corruption.

The Quorum setting in the policy will only have an effect if the following items are true:

- The group members are also cluster members.

- GroupName.WAS_CLUSTER=*clustername* must be specified as a property in the group name of any high availability group matching this policy.

When enabled, any group using this policy will not achieve quorum until a majority of the members are running. For example, if there are n members in the group, (n/2) + 1 servers must be online in order to achieve quorum. No group members will be activated until quorum has been achieved.

The quorum mechanism is designed to work in conjunction with a hardware control facility that allows application servers to be shut down if a failure causes the group to be partitioned.

Additional Properties

Specifies one or more of the following options, depending on the type of policy you selected:

Custom properties	Click to specify custom properties for the policy.
Match criteria	Click to set up a match criterion for the policy.
Preferred servers	Click to set up a list of servers that are given preference when group members are activated.
Static group servers	Click to set up a list of the specific servers that are activated.

New core group policy definition

Use this page to create a new policy for a high availability group.

When you create a new policy, the first page that displays lets you select a policy type. To view the administrative console page where you select a policy type, click **Servers > Core Groups > Core group settings > New** or select an existing core group. Then click **Policies > New**.

Select one of the following policy types. For messaging engines in service integration, you must select either the Static, One of N, or the No operation policy.

- All active policy: Under this policy, all of the group members are activated.
- M of N policy: Under this policy, *M* group members are activated. The number represented by *M* is defined as part of the policy details.
- No operation policy: Under this policy, no group members are activated.
- One of N policy: Under this policy, only one group member is activated.
- Static policy: Under this policy, only specified group members are activated.

After selecting a policy, click **Next** to continue.

Preferred servers

Use this page to define the ordered list of *preferred servers* for the selected policy. The policy gives preference to the servers in this list when activating group members.

To view this administrative console page, you must be working with a policy that has a policy type of M of N or One of N. If your policy has one of these policy types, click **Servers > Core Groups > Core group settings > New**, or select an existing core group. Then click **> Policies > New >** , or select an existing policy. In the Additional Properties section, select **Preferred Servers**.

Use **Add** and **Remove** to move servers into and out of the list of preferred servers. Use **Move up** and **Move down** to adjust the order within the list of preferred servers. Make sure that the most preferred server is at the top of the list and the least preferred server is at the bottom.

Click **OK** to make your changes effective. Click **Save** to save and synchronize your changes with all managed nodes.

Changes to the preferred servers list take affect as soon as they are saved and synchronized. You do not have to stop and restart the affected application servers.

Match criteria collection

Use this page to view the match criteria that are defined for a policy.

To view this administrative console page, click **Servers > Core Groups > Core group settings > New**, or select an existing core group, and then select **Policies > New**, or select an existing policy, and then click **Match criteria**.

Click **New** to create a new match criterion for the policy. Click the name of a match criterion to change any of that criterion's properties.

Name

Specifies the name portion of a name-value pair that is part of the name of the high availability group that you are associating with this policy.

Value

Specifies the value portion of a name-value pair that is part of the name of the high availability group that you are associating with this policy.

Description

Specifies a description of the match criterion. Make the description meaningful. For example, the description might indicate the high availability group that this name-value pair matches.

Match criteria settings

Use this page to define a match criterion for a policy.

To view this administrative console page, click **Servers > Core Groups > Core group settings > New**, or select an existing core group, and then click **Policies > New**, or select an existing policy. Finally, click **Match criteria >** *criterion name*.

The name and value fields should match a name-value attribute included in the name of a high availability group you want associated with this policy.

After you define a match criterion, click **Apply** to make your changes effective. Click **Save** to save and synchronize your changes with all managed nodes.

Name

Specifies the name portion of a name-value pair that is part of the name of the high availability group that you are associating with this policy.

Value

Specifies the value portion of a name-value pair that is part of the name of the high availability group that you are associating with this policy.

Description

Specifies a description of the match criterion. Make the description meaningful. For example, the description might indicate the high availability group that this name-value pair matches.

Static group servers collection

Use this page to designate for a static policy which high availability group members should be made active.

This option is available under Additional Properties only if Static is selected as the policy type. To view this administrative console page, click **Servers > Core Groups > Core group settings >** *core_group_name* **> Policies >** *static_policy_name* **> Static group servers**.

Use **Add** and **Remove** to move servers into and out of the list of servers that should be activated. Only high availability group members that are associated with this policy appear on this page.

After you finish updating the list, click **Apply** to make your changes effective. Click **Save** to save and synchronize your changes with all managed nodes.

Chapter 7. Selecting the policy for a high availability group

Every high availability group has an associated policy. The high availability manager uses this policy to determine which members of a high availability group to put in the active state.

Before you begin

Before you select a policy for a high availability group, you should review the following topics:

-
- "High availability group policy modification guidelines" on page 35
- "High availability group policies" on page 29

You should also know:

- The name of the core group that you want to associate with the new policy.
- The name of the high availability group that you want this policy to control.
- The function, such as transaction log recovery or messaging engine, that is associated with this high availability group.
- The policy types, such as One of N or Static, that this function supports.
- The type of policy you want to create.
- The policy settings, such as failback, and preferred servers only, that you want to configure for this policy.

About this task

You have multiple policies defined for a high availability group, and you want to specify which of these policies the high availability manager uses to govern the group.

To select a policy for a high availability group:

Procedure

1. In the administrative console, click **Servers > Core Groups > Core group settings >** *core_group_name*.
2. Click the **Runtime** tab to determine both the name of the high availability group, and the name of the policy that is currently controlling the group. See Chapter 5, "Viewing high availability group information," on page 39 for more information on how to perform this step. You must have at least one of the group members running.
3. Click the **Configuration** tab to determine the match criteria defined in the current high availability group policy.
4. Use the information you obtained in the previous steps and update the match criteria for the policy you are selecting. The match criterion must contain all of the match criteria from the original policy, and at least one additional attribute from the name of the high availability group.
5. Click **OK** and then click **Review**.
6. Select **Synchronize changes with nodes**, and then click **Save**.

Results

The high availability manager uses the new policy to govern the designated high availability groups.

Chapter 8. Creating a new core group (high availability domain)

A default core group, called DefaultCoreGroup, is created for each cell. This default core group or high availability domain as it is sometimes referred to, is sufficient in most configurations. However there are some circumstances under which you need to create additional core groups for a cell.

Before you begin

Determine how to segment your existing cell into multiple core groups. Use the following rules as guidelines:

- All members of a cluster must be in the same core group. A core group can contain multiple clusters.
- Core group members cannot cross firewall boundaries.
- Put clusters with direct relationships in the same core group. Examples of direct relationships include:
  - A single application is deployed on multiple clusters.
  - An application on one cluster calls an application on another cluster.

bprac: You should also try to keep your core groups homogeneous. If your installation topology requires you to set up a mix of processes that use and do not use the high availability manager, you can:

- Create a new core group and move all application servers for which the high availability manager is disabled to this core group. A core group that is not managed by a high availability manager does not have a size limit.
- Leave the remaining applications servers that require high availability manager services in the default core group. If the number of application servers remaining in the default core group is too large for the high availability manager to handle efficiently, create another core group, and move some of these application servers into that core group.

About this task

You might want to add another core group to a cell if you are scaling up the number of servers in a cell. Multiple core groups are recommended for a large cell.

To create a new core group:

Procedure

1. In the administrative console, click **Servers > Core Groups > Core group settings > New** .
2. In the Name field, specify a unique name for the new core group. The name can contain alpha and numeric characters, but not the following special characters:

   ```
   # \ / , : ; " * ? < > | = + & % '
   ```

   The name also cannot begin with a period (.) or a blank space. A blank space does not generate an error. However, leading and trailing blank spaces are automatically deleted from the name.
3. Add a description of this core group that helps other administrators understand the purpose of this core group.
4. Click **OK** and then click **Review**.
5. Select **Synchronize changes with nodes**, and then click **Save**.

Results

The cell contains another core group.

What to do next

You must now:

- Complete your core group configuration. The initial core group settings and policies are derived from a template. If the settings from the default template do not meet your requirements, you can:
  - Change the number of coordinators for this core group.
  - Change the transport type for this core group.
  - Add policies for this core group.
- Move members to the new core group.
- Create bridges between core groups. If clusters with direct relationships are not in the same core group, set up a core group bridge to connect the related core groups.

.

Specifying a core group when adding a node

By default, a cell contains a single core group. In this situation, during node federation, a node agent is created and automatically added to this core group. However, if the cell contains multiple core groups, you must specify the core group to which the node agent is assigned.

Before you begin

If the cell to which you are adding a node agent contains multiple core groups, determine the core group to which you want this node agent to belong.

About this task

You have a cell that contains multiple core groups and you want to select the core group to which a newly created node agent is added.

To add a newly created node agent to a specific core group:

Procedure

Include the coregroupname parameter on the **addNode** command. For example:

```
addNode dmgr_host dmgr_port  -coregroupname existing_core_group_name
```

Results

The node agent resides in the specified core group.

Specifying a core group when creating an application server

By default, a cell contains a single core group. In this situation, whenever you add an application server to that cell, it is automatically added to this core group. However if the cell contains multiple core groups, you must specify the core group to which you want the application server assigned.

Before you begin

If the cell to which you are adding an application server contains multiple core groups, determine the core group to which you want this application server to belong.

About this task

You have a cell that contains multiple core groups and you want to select the core group to which a newly created application server is added.

To add a newly created node agent to a specific core group:

Procedure

1. In the administrative console, click **Servers > Server Types > WebSphere application servers > New** .
2. Select the node on which you want to create the application server.
3. Specify a name for the new application server, and then click **Next**.
4. Select the server template that you want for this application server, and then click **Next**. Usually, you want to use the template for the default server.
5. Select the core group to which you want this application server to belong from the list of available core groups, and then click **Next**.
6. Confirm the selections on the summary presented, and click **Finish**.
7. Click **OK**, and then click **Save** to save your changes.
8. Select **Synchronize changes with nodes** and click **Save** again.
9. Restart all members of the core group.
10. Start the server.

Results

After the server starts, it becomes a member of the selected core group.

Chapter 9. Configuring core groups

Configuring core group preferred coordinators

The high availability manager uses an ordered list of preferred core group servers when it selects servers to host the coordinators. If the default list is inappropriate, you can change the list such that other servers are selected as coordinators.

Before you begin

Use the following guidelines to determine the appropriate ordered list of preferred coordinators:

- Make sure that you are familiar with the content of the *Core group coordinator* topic.
- Select a set of servers that infrequently start and stop, are stable, and that are located on capable systems with large amounts of system memory and a fast processor.

bprac: Even though it is possible to use a deployment manager as a core group coordinator, it is recommended that you use an application server that is not a deployment manager.

About this task

You might want to change the ordered list of preferred core group servers if:

- The system where the default coordinator is located has insufficient resources or capacity to serve as a coordinator.
- A server that normally gets selected as the coordinator starts and stops frequently.

To change the ordered list of preferred core group servers:

Procedure

1. In the administrative console, click **Servers** > **Core Groups** > **Core group settings** and select an existing core group.
2. Click **Preferred coordinator servers** and then add or delete the appropriate application servers from the list of preferred coordinator servers. Use **Add** and **Remove** to move servers into and out of the list of core group servers on which the coordinator service can run. Use **Move up** and **Move down** to adjust the order of the servers within this list. Make sure that the most preferred server is at the beginning of the list and the least preferred server is at the end.
3. Click **OK** and then click**Review**.
4. Select **Synchronize changes with nodes**, and then click **Save**.

Results

Your changes take effect immediately after synchronization completes. The members of the core group pick up these changes dynamically.

Preferred coordinator servers settings

Use this page to define an ordered list of preferred core group servers. This list indicates where the high availability manager coordinators will run.

To view this administrative console page, click **Servers** > **Core Groups** > **Core group settings** > **New**, or select an existing core group, and then click **Preferred coordinator servers**.

Use **Add** and **Remove** to move servers into and out of the list of core group servers on which coordinator service will run. Use **Move up** and **Move down** to adjust the order within this list. Make sure that the most preferred server is at the top of the list and the least preferred server is at the bottom.

Click **OK** to make your changes effective. Click **Save** to save and synchronize your changes with all managed nodes.

Changing the number of core group coordinators

For a large cell, multiple coordinators are recommended.

Before you begin

Before you change the number of core group coordinators for a cell:

- Make sure that you are familiar with the content of the topic *Core group coordinator*.
- Determine the number of coordinators for the core group. A general guideline is approximately one coordinator per 20 core group members.

About this task

Change the number of core group coordinators only if:

- IBM support instructs you to do so.
- You are directed to do so as part of the installation process for another WebSphere product.
- The coordinator process is consuming abnormally high amounts of memory or CPU resources.
- You are scaling up the number of servers in a cell.

To change the number of core group coordinators for a core group:

Procedure

1. In the administrative console, click **Servers** > **Core Groups** > **Core group settings** > , and select an existing core group.
2. In the **Number of coordinators** field, specify the number of coordinators that you want for this core group.
3. Click **OK**, and then click **Save**.
4. Click **Synchronize changes with nodes** and then click **Save** again to save your changes.
5. If you need to change the list of preferred coordinator servers, click **Preferred coordinator servers** and then add or delete the appropriate application servers from the list of preferred coordinator servers.
6. Click **OK** and then click**Review**.
7. Select **Synchronize changes with nodes**, and then click **Save**.

Results

Your changes take effect immediately after synchronization completes. The members of the core group pick up these changes dynamically.

Core group settings

Use this page to create a core group or to edit an existing core group. A core group is a component of the high availability manager function. It can contain stand-alone servers, cluster members, node agents, administrative agents, and the deployment manager. A core group must contain at least one node agent, administrative agent, or the deployment manager.

Before you create a core group you must understand the relationship of core groups in a high availability environment and know how you intend to use each core group.

To view this administrative console page, click **Servers > Core Groups > Core group settings**. Then, either select an existing core group or click **New**.

On the **Configuration** tab, you can edit fields. On the **Runtime** tab, you can look at read-only information.

After you specify your core group settings, click **Apply** before defining additional properties or setting up a core group bridge.

Name

Specifies the name of the core group. This field can only be edited when you create new core groups.

If you are defining a new core group, specify a name that is unique among the existing core groups. It is useful to other product administrators if the name helps to define the use of this core group, and if it is consistent with the names of the other core groups in the cell.

This field can contain alpha and numeric characters. The following characters cannot be used in this field:

`# \ / , : ; " * ? < > | = + & % '`

Also, the name cannot begin with a period (.) or a blank space. A blank space does not generate an error. However, leading and trailing blank spaces are automatically deleted.

For example, `DefaultCoreGroup` is the name of the core group that contains the deployment manager server process.

Description

This field is optional. If used, it should provide useful information about the core group. You can use this field to help describe the purpose of a core group or to provide important information about a core group. For example, the value specified for **Description** for the default core group DefaultCoreGroup, that is provided with the product, is `The default core group cannot be deleted.`.

The supported length of this field is large enough to accommodate long descriptions. However, long descriptions take time to load and can cause a delay when displaying the page.

Number of coordinators

Specifies the number of coordinators for this core group. The coordinator is the aggregation point for high availability manager information. The coordinator determines group membership and communicates state and status to the other members of the core group.

The default value is one coordinator, although multiple coordinators are advisable for large core groups. All of the data for a core group must fit in the memory of the allocated coordinators. Therefore, if you have multiple coordinators, more memory is available to store the core group data. If a single coordinator is defined in a system with a large core group, the core group data might consume all of that coordinators memory, causing processing errors.

Transport memory size

Specifies, in megabytes, the maximum size of dynamically allocated memory that the transport can acquire. The range is from `10` to `2147483647` megabytes. However, this value cannot exceed the maximum heap size that is specified for the Java virtual machine. When this limit is reached, congestion control is invoked, and communication between core group members is limited, and processing errors might occur.

Transport type

Specifies the transport mechanism to use for communication between members of a core group. You must select either **Channel framework** or **Unicast**.

gotcha: If a core group needs to use the core group bridge service, you must select **Channel framework** as the transport mechanism for that core group. If you select Unicast, you might receive error message CHFW0029E, which indicates that the transport chain could not be initialized because the address was already in use.

depfeat: Support for unicast transports is deprecated. Select **Channel framework** as your transport type if your environment makes it possible to do so.

Transport chain

Specifies the name of the transport chain, if you select channel framework for the transport type.

Additional Properties

Core group servers
Specifies the server processes that belong to the core group. Server processes include the deployment manager, node agents, application servers, and cluster members. You can use this panel to move server processes to a different core group.

Custom properties
Specifies the custom properties that are used for configuration purposes.

Policies
Specifies the policies that determine which members of a high availability group are made active.

Preferred coordinator servers
Specifies which core group servers are preferred coordinator servers.

Related Items

Core group bridge settings
Click to navigate to the administrative console page where you can define a core group bridge. A core group bridge is required to establish communication between core groups that reside in different cells.

Group name properties

Specifies one or more name=value pairs as the match criterion for a high availability group. If you specify more than one name=value pair, use a comma to separate the pairs. You can specify an asterisk (*) to obtain the selected information for all of the high availability groups within this core group.

gotcha: You can specify multiple properties as the value in the **Group name properties** field. However, if you specify multiple properties, commas must be used to separate the properties. Blank spaces are not allowed in a value specified in this field.

When a component creates a new high availability group, it establishes a map of the properties for that group. This map becomes the group name, and is used to uniquely identify that high availability group.

After you specify a match criterion or an asterisk, complete the following actions:

- Select **Calculate groups** to determine how many high availability groups have names that match this criterion. The number of high availability groups that are found that have the specified group name properties displays in the **Number of matches field**.
- Select **Show groups** to view a list of the currently running high availability groups that match this criterion. This list indicates the following information for each group:
  - Its high availability group name
  - Whether or not quorum has been enabled
  - The policy that is associated with the high availability group. If more than one policy is listed for a high availability group, you must change the match criterion for one or more of your policies so that only one policy is associated with this high availability group.

  - Its status. If only one policy is listed in the Policy column, the OK icon is displayed in the Status column. If more than one policy is listed in the Policy column, the Error icon is displayed in the Status column.
- Select **Show severs** to view a list of the servers that are hosting active members of the high availability groups that match the value specified for the **Group name properties** field. For each server, this list indicates:
  -
  - The name of the node on which these servers reside
  - The version of the product on which these servers run
  - The number of high availability group members that are currently active on these servers

For example, suppose that the following high availability groups are defined for a core group:

- Component A uses the following properties for its group name: [name=compA, policy=oneofN, owner=smith]
- Component B uses the following properties for its group name: [name=compB, policy=MofN, owner=smith]
- Component C uses the following properties for its group name: [name=compC, policy=oneofN, owner=smith]

If you specify `policy=oneofN` in the **Group name properties** field and then select **Show groups**, the groups for components A and C are listed.

If you specify `owner=smith` in the **Group name properties** field and then select **Show groups**, the groups for components A, B and C are listed.

If you specify `name=compC,policy=oneofN,owner=smith`, which is all of the name properties for component C, in the **Group name properties** field and then select **Show groups**, only the group for component C is listed.

Core group custom properties

Use these custom properties for advanced configurations for core groups.

Note: This topic references one or more of the application server log files. Beginning in WebSphere Application Server Version 8.0 you can configure the server to use the High Performance Extensible Logging (HPEL) log and trace infrastructure instead of using `SystemOut.log` , `SystemErr.log`, `trace.log`, and `activity.log` files or native z/OS logging facilities. If you are using HPEL, you can access all of your log and trace information using the LogViewer command-line tool from your server profile bin directory. See the information about using HPEL to troubleshoot applications for more information on using HPEL.

To configure a custom property, on the indicated Custom properties page, complete one of the following actions:

- If the custom property is in the list of defined custom properties, click on that property and then enter an appropriate value in the Value field.
- If the custom property is not in the list of defined custom properties, click **New** and then enter the name of the custom property in the **Name** field and an appropriate value in the **Value** field.

Note: When you set the core group custom properties, make sure that the core group or the cell is in a stable state. You do not want to set these custom properties when you are simultaneously starting or stopping servers. Also, ensure that the high availability manager is in a known, good state. With these best practices in mind, consider the following best practices for some specific core group custom properties:

- When you set the IBM_CS_WIRE_FORMAT_VERSION custom property to 6.1.0, the value is automatically detected and applied to all nodes. You do not need to stop or start servers. For information on the custom property, see the “IBM_CS_WIRE_FORMAT_VERSION” on page 68 custom property description.
- Although you can set the IBM_CS_HAM_PROTOCOL_VERSION custom property value in a topology in which only a single core group exists, it does not provide any benefit. However, if you need to later partition a cell into multiple core groups and bridge them together, it is better to have the core groups running the most current possible protocol version before you start the bridges. If your topology contains multiple core groups that are bridged together, set this value to the most current possible version. For the applicable values and information on how to set this custom property, see the IBM_CS_HAM_PROTOCOL_VERSION custom property description.

You can use the custom properties page to define the following core group custom properties:

- “IBM_CS_DATASTACK_MEG”
- “IBM_CS_FD_CONSECUTIVE_MISSED” on page 65
- “IBM_CS_FD_PERIOD_SECS” on page 65
- “IBM_CS_HAM_PROTOCOL_VERSION” on page 66
- “IBM_CS_IP_REFRESH_MINUTES” on page 66
- “IBM_CS_SOCKET_BUFFER_SIZE” on page 67
- “IBM_CS_UNICAST_DISCOVERY_INTERVAL_SECS” on page 67
- “IBM_CS_WIRE_FORMAT_VERSION” on page 68

IBM_CS_DATASTACK_MEG

Use this custom property to eliminate a condition that is reported by a message that is displayed repeatedly in your `SystemOut.log` file.

Note: This topic references one or more of the application server log files. Beginning in WebSphere Application Server Version 8.0 you can configure the server to use the High Performance Extensible Logging (HPEL) log and trace infrastructure instead of using `SystemOut.log` , `SystemErr.log`, `trace.log`, and `activity.log` files or native z/OS logging facilities. If you are using HPEL, you can access all of your log and trace information using the LogViewer command-line tool from your server profile bin directory. See the information about using HPEL to troubleshoot applications for more information on using HPEL.

depfeat: This custom property is deprecated. Do not use this custom property unless you are running in a mixed cell environment that includes at least one core groups that contains a mixture of Version 7.0 and Version 6.x processes.

mixv: If you are running in a mixed cell environment, and you have core groups that contain a mixture of Version 7.0 and Version 6.x processes, you must continue to use the IBM_CS_DATASTACK_MEG core group custom property to change how frequently the Discovery Protocol attempts to open a new network.

You might see a message, similar to the following message, repeated multiple times in the SystemOut.log file:

```
[9/24/04 13:28:19:497 CDT] 00000013 VSync         W
DCSV2005W: DCS Stack DefaultAccessPointGroup.P at Member 172.16.1.86:9353:
The amount of memory available for synchronization is low. The configured memory
size is 418816 bytes. Currently used memory size is 420307 bytes.
```

If the member IP address is in the format of a dotted decimal IP address and port, you can eliminate these messages by increasing the amount of memory that is allocated to the core stack that is used for core group communication. Increase the value of this property until you no longer see the message in your `SystemOut.log` file. Because the memory is dynamically allocated, setting a larger stack size than you need does not cause memory problems.

Note: You can also set this custom property:

- On the core group bridge interface that contains the particular core group member that is in the messages.
- On the access point group or the core group access point for the particular core group member that is in the messages.

See “Core group bridge custom properties” on page 127 for more information about how to set the property at those levels.

Table 3. Units and default. The following table describes the units and default for the setting.

Units	megabytes
Default	5

To set a value for this custom property, in the administrative console, click **Servers > Core Groups > Core group settings >** *core_group_name*. Then, under Additional Properties, click **Custom properties**.

IBM_CS_FD_CONSECUTIVE_MISSED

This custom property is used to specify the consecutive number of heartbeats that must be missed before the core group member is considered failed.

depfeat: This custom property is deprecated. Do not use this custom property unless you are running in a mixed cell environment that includes at least one core groups that contains a mixture of Version 7.0 and Version 6.x processes.

However, you can also use the Discovery and failure detection page in the administrative console to change the settings for the Discovery and Failure Detection protocols. Unless you are running in a mixed cell environment, using the administrative console is the preferred technique for changing the settings for the Discovery and Failure Detection protocols because the IBM_CS_FD_CONSECUTIVE_MISSED property is deprecated.

mixv: If you are running in a mixed cell environment, and you have core groups that contain a mixture of Version 7.0 and Version 6.x processes, you must continue to use the IBM_CS_FD_CONSECUTIVE_MISSED core group custom property to change how many consecutive heartbeats a core group member must miss before that core group member is considered failed.

Table 4. Units and default. The following table describes the units and default for the setting.

Units	seconds
Default	6

To set a value for this custom property, in the administrative console, click **Servers > Core Groups > Core group settings >** *core_group_name*. Then, under Additional Properties, click **Custom properties**.

IBM_CS_FD_PERIOD_SECS

This custom property is used to change how frequently the Failure Detection Protocol checks the core group network connections that the discovery protocol establishes. The Failure Detection Protocol notifies the Discovery Protocol if a connection failure occurs.

depfeat: This custom property is deprecated. Do not use this custom property unless you are running in a mixed cell environment that includes at least one core groups that contains a mixture of Version 7.0 and Version 6.x processes.

However, you can also use the Discovery and failure detection page in the administrative console to change how frequently the high availability manager Discovery Protocol checks for new core group members. Unless you are running in a mixed cell environment, using the administrative console is the preferred technique for updating this setting because the IBM_CS_FD_PERIOD_SECS property is deprecated.

mixv: If you are running in a mixed cell environment, and you have core groups that contain a mixture of Version 7.0 and Version 6.x processes, you must continue to use the IBM_CS_FD_PERIOD_SECS core group custom property to change how frequently the Failure Detection Protocol checks the core group network connections that the discovery protocol establishes.

Table 5. Units and default. The following table describes the units and default for the setting.

Units	seconds
Default	6

To set a value for this custom property, in the administrative console, click **Servers > Core Groups > Core group settings >** *core_group_name*. Then, under Additional Properties, click **Custom properties**.

IBM_CS_HAM_PROTOCOL_VERSION

Use this custom property to configure all of the members of the selected core group to run with a different version of the higher level core group protocol, which is also referred to as the high availability manager protocol. You can change the value of this property while the core group members continue to run. After you save and synchronize the changed value to all of the nodes containing core group processes, the high availability manager automatically detects the configuration change and starts using the new core group protocol version to communicate with the core group members.

Table 6. Version ID and description. The following table lists the supported core group protocol version IDs.

Version ID	Required Minimum Product Level	Description
6.0.2.31	6.0.2.31 for Version 6.0, 6.1.0.19 for Version 6.1, 7.0.0.1 for Version 7.0	This protocol version is the original or base version of the HAM protocol. This version was included in the 6.0.2.31, 6.1.0.19, and 7.0.0.1 service packs to facilitate core group bridge scalability. This version is recommended for topologies that contain multiple core groups and core group bridges as part of their configuration.

Set this custom property to the value in the Version ID column of the previous table. To set this custom property, complete the following process:

1. Stop the core group bridges.
2. Set the custom property on all core groups.
3. Restart the high availability coordinators in all core groups.
4. Start the core group bridges.

IBM_CS_IP_REFRESH_MINUTES

Use this custom property to adjust how frequently the core group IP cache is cleared.

The caching of name-to-IP address information at the core group level eliminates some of the system overhead required to assign IP addresses to core group members.

Table 7. Units, default, and acceptable values. The following table describes the units, default, and acceptable values for the setting.

Units	seconds
Default	60
Acceptable values	Any positive integer. 1 is the minimum value that can be specified

To set a value for this custom property, in the administrative console, click **Servers > Core Groups > Core group settings >** *core_group_name*. Then, under Additional Properties, click **Custom properties**.

IBM_CS_SOCKET_BUFFER_SIZE

Use this custom property to change the size of the socket buffer that the core group transport obtains.

"Configuring core group socket buffers" on page 83 includes a table that shows the relationship between the values that can be specified for this property and the underlying memory allocation size per socket buffer type.

Table 8. Units and default. The following table describes the units and default for the setting.

Units	One of the following: • `No over rides` • `small` • `medium` • `large`
Default	2 megabytes for all buffer types.

To set a value for this custom property, in the administrative console, click **Servers > Core Groups > Core group settings >** *core_group_name*. Then, under Additional Properties, click **Custom properties**.

IBM_CS_UNICAST_DISCOVERY_INTERVAL_SECS

This custom property is used to change how frequently the high availability manager Discovery Protocol checks for new core group members. A new core group member is not able to communicate with other core group members until the Discovery Protocol establishes communication between the new member and the existing members.

depfeat: This custom property is deprecated. Do not use this custom property unless you are running in a mixed cell environment that includes at least one core groups that contains a mixture of Version 7.0 and Version 6.x processes.

However, you can also use the Discovery and failure detection page in the administrative console to change how frequently the high availability manager Discovery Protocol checks for new core group members. Unless you are running in a mixed cell environment, using the administrative console is the preferred technique for updating this setting because the IBM_CS_UNICAST_DISCOVERY_INTERVAL_SECS property is deprecated.

mixv: If you are running in a mixed cell environment, and you have core groups that contain a mixture of Version 7.0 and Version 6.x processes, you must continue to use the IBM_CS_UNICAST_DISCOVERY_INTERVAL_SECS core group custom property to change how frequently the Discovery Protocol attempts to open a new network.

Table 9. Units and default. The following table describes the units and default for the setting.

Units	seconds
Default	30

To set a value for this custom property, in the administrative console, click **Servers > Core Groups > Core group settings >** *core_group_name*. Then, under Additional Properties, click **Custom properties**.

IBM_CS_WIRE_FORMAT_VERSION

Use this custom property to configure all of the members of the selected core group to run with a different version of the core group protocol. You can change the value of this property while the core group members continue to run. After you save and synchronize the changed value to all of the nodes containing core group processes, the high availability manager automatically detects the configuration change and starts using the new core group protocol version to communicate with the core group members.

depfeat: Protocol versions 6.0.0 and 6.0.2.9 are deprecated. Whenever possible you should use a newer protocol version.

Table 10. Supported core group protocol version IDs. The following table lists the supported core group protocol version IDs. 6.0.0 is the default value.

Version ID	Required Minimum Product Level	Description
6.0.0	6.0.0.0 for Version 6.0, 6.1.0.0 for Version 6.1, 7.0.0.0 for Version 7.0	This protocol version is the original or base version. All versions of the high availability manager can use this protocol version. If you do not specify a particular protocol version the high availability manager uses this version.
6.0.2.9	6.0.2.9 for Version 6.0.2, 6.1.0.0 for Version 6.1, 7.0.0.0 for Version 7.0.	This protocol version was included in the 6.0.2.9 service pack, and in Versions 6.1, and 7.0 of the product to facilitate core group bridge scalability. This version is recommended for large topologies that contain multiple core groups and core group bridges as part of their configuration.
6.1.0	6.1.0.0 for Version 6.1, 7.0.0.0 for Version 7.0	This protocol version was included in Versions 6.1, and 7.0 of the product to add core group scalability improvements, and more support for large topologies.

You can set this custom property to the values that are listed in the Version ID column of the previous table. Set the value to the most current value that is possible for your environment.

To set a value for this custom property, in the administrative console, click **Servers > Core Groups > Core group settings >** *core_group_name*. Then, under Additional Properties, click **Custom properties**.

Configuring the default Discovery Protocol for a core group

The default Discovery Protocol for a core group establishes network connectivity between a new core group member and the other members of that core group. Perform this task if you need to tune the default Discovery Protocol behavior for a core group. The default value of 60 seconds, which is set by the product installation process, provides an acceptable process detection time for most situations.

Before you begin

- Understand the concepts that are described in the topic *Core group discovery and failure detection protocols*.
- Verify that you are using the default Discovery and Failure Detection providers. If you are using an alternative protocol provider, do not perform the steps in this task because these steps only apply if you are using the default discovery and failure detection providers.
- Determine how long you want the default Discovery Protocol to wait before it recalculates the set of unconnected core group members, and attempts to open connections to those members.

About this task

Use the administrative console or the wsadmin tool to adjust this setting if it is not appropriate for your environment unless you are running in a mixed cell environment that includes core groups that contain a mixture of Version 7.0 and Version 6.x processes,

mixv: If you are running in a mixed cell environment, and you have core groups that contain a mixture of Version 7.0 and Version 6.x processes, you must continue to use the IBM_CS_UNICAST_DISCOVERY_INTERVAL_SECS core group custom property to adjust how frequently the default Discovery Protocol attempts to open a new network. To specify this custom property:

1. In the administrative console, click **Servers > Core Groups > Core group settings >** *core_group_name*. Then, in the Additional Properties section, click **Custom properties > New**.
2. In the **Name** field, specify `IBM_CS_UNICAST_DISCOVERY_INTERVAL_SECS`, and specify a new time interval in the **Value** field.

To use the administrative console to change the wait interval for the default Discovery Protocol, complete the following steps.

Procedure

1. In the administrative console, click **Servers > Core Groups > Core group settings >** *core_group_name*.
2. In the Additional Properties section, click **Discovery and failure detection**. The **Use the default protocol providers** option must be selected. If this option is not selected, do not perform any more of the steps in this task.
3. Specify a new value for the **Discovery period** property.

   Decreasing this time interval might improve the detection of core group members.
4. Click **OK** and then click **Review**.
5. Select **Synchronize changes with nodes**, and then click **Save**.
6. Restart all members of the core group.

Results

After the servers restart, the core group members all run with the new Discovery Protocol setting.

Discovery and failure detection settings

Use this page to configure the discovery and failure detection settings for a core group. These settings are used to monitor the health of core group members. The discovery protocol establishes network connectivity between core group members of the core group. The failure detection protocol monitors the established network connections. Both the default discovery protocol and the default failure protocol run at regularly scheduled intervals on all started core group members.

To view this administrative console page, click **Servers > Core Groups > Core group settings >** *core_group_name*. Then, in the Additional Properties section, click **Discovery and failure detection**.

Use the default protocol providers

Select this option if you want to use the default Discovery Protocol and default Failure Detection protocol that are provided with the product.

Discovery period

Specifies, in seconds, the time interval that the default Discovery Protocol waits before it recalculates the set of unconnected core group members, and attempts to open connections to those members.

Decreasing this value might improve the detection of core group members. However, decreasing this value will also cause the default discovery protocol to consume additional system resources, which might impact performance.

You can specify a value for this property only if you select the **Use the default protocol providers**

The default value is 60 seconds.

Heartbeat transmission period

Specifies, in milliseconds, the amount of time that elapses between failure detection heartbeats. The default Failure Detection Protocol sends out a heartbeat once during the specified time limit to determine whether a core group member is active. Increasing the length of time between heartbeats might decrease the use of system resources such as CPU. However, decreasing the length of time between heartbeats might improve the detection of failed core group members.

You can specify a value for this property only if you select the **Use the default protocol providers**

The default value is 30000 milliseconds.

Heartbeat timeout period

Specifies, in milliseconds, the amount of time that constitutes a heartbeat timeout. This value must be an integer multiple of the value specified for the **Heartbeat transmission period** property. For example, if you set the Heartbeat transmission period property to 30000, the value you specify for the Heartbeat timeout period property must be n x 30000, where n is a positive integer.

You can specify a value for this property only if you select the **Use the default protocol providers**

The default value is 180000 milliseconds.

Use alternative protocol providers

Select this option if you want to use an alternative protocol provider that is provided with the product instead of the default Discovery Protocol and the default Failure Detection Protocol.

You should select this option only if there is an alternative protocol provider available for the platform on which you are running the product.

Factory class name

If you select the **Use alternative protocol providers** option, you must specify the fully qualified class name of the factory that is used to create the alternate protocol provider.

The fully qualified class name of the factory that is used to create the alternate protocol provider that is available for z/OS is `com.ibm.ws.xcf.groupservices.LivenessPluginZoSFactory`.

If you specify the z/OS Cross-system Coupling Facility (XCF)-based protocol provider as your alternate protocol provider, an XCF group is created that establishes monitoring services for all of the core group members, monitors each group member address space, and notifies the high availability manager whenever a member address space shuts down or terminates.

Selecting an alternate protocol provider for a core group

Perform this task if you want to use an alternate protocol provider instead of the default Discovery Protocol and the default Failure Detection Protocol to monitor and manage communication between core group members. In general, alternate protocol providers, such as the z/OS Cross-system Coupling Facility (XCF)-based provider, use less system resources than the default Discovery Protocol and Failure Detection Protocol, especially during times when the core group members are idle. An alternate protocol provider generally use less system resources because it does not perform the member-to-member TCP/IP pinging that the default protocol providers use to determine if a core group member is still active.

Before you begin

- Understand the concepts that are described in the *Core group Discovery Protocol* and *Core group Failure Detection Protocol* topics.
- Verify that all of the servers that are members of the core group are at Version 7.0 or higher. If the core group contains servers at a version earlier than 7.0, then you must use the default Discovery Protocol and the default Failure Detection Protocol.
- Verify that all of the servers that are members of the core group are running on the same operating system. The alternative protocol provider can only be used in an homogenous environment. If the cell contains any servers that are running on different operating systems, you must use the default Discovery Protocol and default Failure Detection Protocol. For example, you cannot have an environment where some of the members of the core group are running on z/OS and other members are running on a Linux operating system.
- Verify that your system meets all of the system requirements for your alternate protocol provider.

  For example, if you are going to use the z/OS Cross-system Coupling Facility as your alternate protocol provider, you must verify that the z/OS VTAM® component is configured to start with the XCFINIT parameter set to YES. Starting VTAM with XCFINIT=YES specified in ATCSTRxx enables TCP/IP to utilize the services that the z/OS Cross-system Coupling Facility provides. See the z/OS product library for more information about VTAM and the z/OS Cross-system Coupling Facility.

About this task

To select an alternate protocol provider for a core group, complete the following steps.

Procedure

1. In the administrative console, click **Servers > Core Groups > Core group settings >** *core_group_name*.
2. In the Additional Properties section, click **Discovery and failure detection**.
3. Select **Use alternative protocol providers**
4. In the **Factory class name** field, specify the fully qualified class name of the factory that is used to create the alternate protocol provider.

   For example, the fully qualified class name for the alternative protocol provider that enables XCF to establish and monitor communication between members of this core group is com.ibm.ws.xcf.groupservices.LivenessPluginZoSFactory.
5. Click **OK** and then click **Review**.
6. Select **Synchronize changes with nodes**, and then click **Save**.
7. Restart all members of the core group.

Results

After the servers restart, all communication between members of this core group is established and monitored by the selected alternate protocol provider.

Configuring the default Failure Detection Protocol for a core group

The default Failure Detection Protocol monitors the core group network connections that the default Discovery Protocol establishes, and notifies the default Discovery Protocol if a connection failure occurs.

Before you begin

- Understand the concepts that are described in the topic *Core group discovery and failure detection protocols*.
- Check your operating system settings that are relevant to TCP/IP socket closing events.
- Determine your failure detection goals and which settings must change to accomplish these goals.

  The value that you specify for the **Heartbeat timeout period** should equal the product of multiplying the value specified for the **Heartbeat transmission period** property, times the **Number of missed consecutive heartbeats** property.

  – The heartbeat transmission period specifies the frequency at which a core group member sends a heartbeat packet over every established connection. The default value for the heartbeat transmission period is 30 seconds.

  – The heartbeat timeout period specifies the failure detection time. If no packets are received during the specified time period, a failure is declared. The default value for the heartbeat transmission period is 180 seconds.

About this task

You might want to perform this task if:
- You want to change the failover characteristics of your system.
- Your core groups are large and analysis indicates excessive CPU usage is spent monitoring heartbeats.

The heartbeat transmission period and heartbeat timeout period are configurable. Use the administrative console or the wsadmin tool to adjust these settings if the default values are not appropriate for your environment, unless you are running in a mixed cell environment that includes core groups that contain a mixture of Version 7.0 and Version 6.x processes,

mixv: If you are running in a mixed cell environment, and you have core groups that contain a mixture of Version 7.0 and Version 6.x processes, you must continue to use the IBM_CS_FD_PERIOD_SECS and IBM_CS_FD_CONSECUTIVE_MISSED core group custom properties to adjust these settings. To specify these custom properties:

1. In the administrative console, click **Servers > Core Groups > Core group settings >** *core_group_name*. Then, in the Additional Properties section, click **Custom properties > New**.
2. In the**Name** field, specify either `IBM_CS_FD_PERIOD_SECS` or `IBM_CS_FD_CONSECUTIVE_MISSED`, and then specify a new value for these properties in the **Value** field.

   The IBM_CS_FD_PERIOD_SECS custom property specifies how frequently the Failure Detection Protocol checks the core group network connections that the discovery protocol establishes.

   The IBM_CS_FD_CONSECUTIVE_MISSED property specifies the number of consecutive heartbeats that a member can missed before it is communication with that member is discontinued.

Remember, when you use the administrative console or the wsadmin tool to configure the Failure Detection Protocol, you configure the heartbeat transmission period, and the heartbeat timeout period. However if you are use the custom properties to configure the Failure Detection Protocol, you configure the heartbeat transmission period, and the number of missed consecutive heartbeats.

To use the administrative console to change the settings for the default Failure Detection Protocol complete the following steps.

Procedure

1. In the administrative console, click **Servers > Core Groups > Core group settings >** *core_group_name*.

2. Then, in the Additional Properties section, click **Discovery and failure detection**. The **Use the default protocol providers** option must be selected. If this option is not selected, do not perform any more of the steps in this task.
3. Specify, in milliseconds, a new value for the **Heartbeat transmission period** property.

   The default value for this property is 30000 milliseconds, which equals 30 seconds.
4. Specify, in milliseconds, a new value for the **Heartbeat timeout period** property.

   The default value for this property is 180000 milliseconds, which equals 180 seconds.
5. Click **OK** and then click **Review**.
6. Select **Synchronize changes with nodes**, and then click **Save**.
7. Restart all of the members of the core group.

Results

After the servers restart, the core group members all run with the new Failure Detection Protocol settings.

Selecting the version of a core group protocol

There are two major categories or groups of high availability manager protocols. Both of these protocol groups can be independently configured to use newer versions of the protocols.

Before you begin

Before you change the version of a core group protocol for a particular core group member, make sure that:

- You are familiar with the content of the *Core group protocol versions* topic. This topic contains information that is helpful in identifying:
  - Which protocol category you want to configure.
  - The custom property that you must use to configure this protocol category.
- You use the tables that are included in the *Core group protocol versions* topic to determine in which releases or service packs each protocol version was added to the product.
- You verify that each member of the core group is running at a product code level (VRM) that supports the new protocol version. The code level must be equal to or greater than the code level in which this protocol version was added to the product. All core group members with which the new protocol version is associated must support that protocol version.

About this task

Note: This topic references one or more of the application server log files. Beginning in WebSphere Application Server Version 8.0 you can configure the server to use the High Performance Extensible Logging (HPEL) log and trace infrastructure instead of using `SystemOut.log` , `SystemErr.log`, `trace.log`, and `activity.log` files or native z/OS logging facilities. If you are using HPEL, you can access all of your log and trace information using the LogViewer command-line tool from your server profile bin directory. See the information about using HPEL to troubleshoot applications for more information on using HPEL.

The high availability manager can use any supported core group protocol versions. However, to ensure that proper communication continues between all core group members, you must make sure that all member of a particular core group use the same version of a core group protocol.

To change the version of a core group protocol complete the following steps.

Procedure

1. In the administrative console, click **Servers > Core Groups > Core group settings** and select an existing core group.
2. Under Additional Properties, click **Custom properties**.
3. Change the value of either the IBM_CS_WIRE_FORMAT_VERSION or IBM_CS_HAM_PROTOCOL_VERSION custom property.

   If you are updating the lower level protocols, which are also referred to as the lower level wire format protocols, change the value of the IBM_CS_WIRE_FORMAT_VERSION custom property.

   If you are updating the higher level protocols, which are also referred to as the high availability manager protocols, change the value of the IBM_CS_HAM_PROTOCOL_VERSION custom property.

   To update either of these custom properties, complete one of the following substeps:

   a. If the property that you want to update does not already exist, click **New** and then specify either `IBM_CS_WIRE_FORMAT_VERSION` or `IBM_CS_HAM_PROTOCOL_VERSION` in the **Name** field and the appropriate version ID in the **Value** field.

   b. If the property that you need to update already exists, click on the property name, and specify the appropriate version ID in the **Value** field.
4. Click **OK**, and then click **Review**.
5. Select **Synchronize changes with nodes**, and then click **Save**.

Results

After synchronization completes, the high availability manager automatically detects the configuration changes and starts to use the new core group protocol version with these core group members.

What to do next

You can check your SystemOut.log file or, for z/OS, either SYSOUT or SYSPRINT, for DCSV0005I and HMGR0226I messages.

Whenever the value specified for the IBM_CS_WIRE_FORMAT_VERSION custom property changes, a DCSV0005I message, similar to the following example, is sent to the SystemOut.log file, or, for the z/OS platform, to either SYSOUT or SYSPRINT:

```
DCSV0005I: DCS Stack DefaultCoreGroup at Member
   DIAMONDCell01\DIAMONDCellManager01\dmgr: Started.
   Stack version information: DCSBV_WAS6_1_20060409.
   Stack protocol information: 61002.
```

The Stack protocol information value at the end of this message indicates the protocol version that is being used. For example, a value of 51001 means the 6.0.0 version is being used. A value of 61002 means the 6.1.0 version is being used.

Whenever the value specified for the IBM_CS_HAM_PROTOCOL_VERSION custom property changes, a HMGR0226I message, similar to the following message, is sent to the SystemOut.log file, or, for the z/OS platform, to either SYSOUT or SYSPRINT:

```
HMGR0226I: The core stack configuration parameter IBM_CS_HAM_PROTOCOL_VERSION
   has been set to 6.0.2.31.
```

This message indicates that the high availability manager protocol version 6.0.2.31 is being used.

Configuring core group memory utilization

You can use the administrative console to control the maximum amount of heap memory that is available for the underlying core group transport to allocate.

Before you begin

You should understand the following information:

- Other factors, such as the number of network interface cards on a machine, how the Network interface card is used, and network speed, can affect the memory that the high availability manager requires to efficiently handle network messages
- The amount of memory that the high availability manager controls while sending network messages varies. Typically, the high availability manager only retains control of this memory for very short lengths of time.
- The default setting of 100 MB is typically sufficient for most network deployment topologies.
- The maximum value for this setting is 250 MB.
- In core groups that contain both Version 6.x and Version 7 processes, all memory settings must be equal. The default memory setting for Version 6.x processes is 10 MB. Therefore you must adjust either the Version 6.x or the Version 7 setting to make both settings equal.

About this task

Note: This topic references one or more of the application server log files. Beginning in WebSphere Application Server Version 8.0 you can configure the server to use the High Performance Extensible Logging (HPEL) log and trace infrastructure instead of using `SystemOut.log` , `SystemErr.log`, `trace.log`, and `activity.log` files or native z/OS logging facilities. If you are using HPEL, you can access all of your log and trace information using the LogViewer command-line tool from your server profile bin directory. See the information about using HPEL to troubleshoot applications for more information on using HPEL.

The high availability manager consumes its allocated heap memory when it sends interprocess messages on behalf of other services that use the high availability manager functionality. For example, the heap memory might be consumed while sending memory-to-memory replication data, or highly available routing data from one core group member to another core group member.

The value that you specify for the Transport memory size property acts as a flow control mechanism that might affect the speed at which data is replicated or the speed at which routing data is made available to other core group members.

Therefore, you might want to perform this task if you are seeing large numbers of any of the following Distribution and Consistency Services (DCS) congestion messages in your `SystemOut.log` file:

```
DCSV1051W, a high severity congestion event for outgoing messages
DCSV1052W, a medium severity congestion event for outgoing messages
DCSV1054W, a medium severity congestion event for incoming messages
```

Under extreme workloads, these messages might still occur on a properly tuned system.

mixv: If you are running in a mixed cell environment, you must continue to use the following two memory configuration attributes to configuring core group memory utilization for the Version 6.x cells:

- The Transport buffer size property, which is set for each individual core group member. To specify a value for this property, in the administrative console, navigate one of the following paths:
  - If the core group member is an application server, click **Servers > Server Types > WebSphere application servers >** *server_name* **> Core group service**.
  - If the core group member is a node agent, click **System Administration > Node agents >** *node_agent_name* **> Core group service**.
  - If the core group member is a deployment manager, click **System Administration > Deployment manager > Core group service**.

- The IBM_CS_DATASTACK_MEG custom property. To specify a value for this custom property, in the administrative console, click **Servers > Core Groups > Core group settings >** *core_group_name* **> Custom properties**.

To change the amount of memory that is available for in-flight messages and network communication buffers, complete the following steps:

Procedure

1. In the administrative console, click **Servers > Core Groups > Core group settings >** *core_group_name*.
2. Change the setting for the Transport memory size property.

   This property, specifies in megabytes, the maximum amount of heap memory that can be allocated to the high availability manager. The maximum value that can be specified for this setting is 250.
3. Click **OK**.
4. Click **Review** to review your changes.
5. Select **Synchronize changes with Nodes**, and then click **Save** to save your changes.
6. Restart all members of the core group.

Results

After the servers restart, they all run with the new memory settings.

Configuring a core group transport

When you configure a core group, you can specify the type of network transport that you want the high availability manager to use for network communications.

Before you begin

Complete the following actions before you start to configure a core group transport.

- Make sure that you are familiar with the content of the topic *Core group transports*.
- Confirm that all node agents in the core group are running.
- Determine which transport type is appropriate for the core group.

About this task

Change the transport setting for a core group if the current transport type does not meet your needs. For example, you might need to increase security, you might want to maximize replication throughput, or a core group might need to use the core group bridge service.

gotcha: If a core group needs to use the core group bridge service, you must select **Channel framework** as its transport mechanism. If you select **Unicast**, you might receive error message CHFW0029E, which indicates that the transport chain was not initialized because the address was already in use.

To change the core group transport for a core group, complete the following procedure.

Procedure

1. In the administrative console, click **Servers > Core Groups > Core group settings**, and select an existing core group, or click **New** if you are creating a new core group.
2. Under Transport type, select the type of transport that you want to use for this core group.

depfeat: You can select one of the following types of transports. However, if possible, select **Channel framework** because support for unicast transports is deprecated.

- **Channel framework** is the default transport type, and is a flexible point-to-point transport. If you choose this transport type, you must also select one of the transport chains that is listed in the **Transport chain** field.
  - If you select **DCS**, the high availability manager performs related network communication over non-secure sockets.
  - If you select **DCS-Secure**, the high availability manager performs related network communication over Secure Sockets Layer (SSL) encrypted sockets.
- **Unicast**, is a point-to-point transport over standard non-secure operating system-level TCP/IP network connection facilities.

3. Click Synchronize changes with nodes, and then click **OK**.
4. Select **Synchronize changes with nodes**, and then click **Save** again.
5. Restart the servers containing core group members.

Results

The new transport values are in effect after the servers restart.

Interoperating with Version 6.0.1.2 processes

The high availability manager supports multihomed hosts, which means that the product processes can communicate with each other even if they are running on different versions of the product. If you are running Version 6.0.1.2 processes that need to communicate with each other, there are high availability manager-related issues that you need to consider.

Before you begin

Know the version levels of the processes that need to communicate with each other. If no V6.0.1.2 processes exist, see “Interoperating with Version 6.0.2 and later processes” on page 79.

About this task

In Version 6.0.1 without Fix Pack 2 installed, an acceptable value for the DCS_UNICAST_ADDRESS host name field is either a host name, for example, myhost.mydomain, or a textual IP address, for example, 192.168.0.2. If a host name is specified, Domain Name Services (DNS) is used to resolve the host name to an IP address. If the host supports multiple IP addresses, where the host name has multiple mappings in DNS, a host name is ambiguous and a textual IP address is required.

After the installation of Version 6.0.1 Fix Pack 2, the asterisk is recognized as a valid value for the DCS_UNICAST_ADDRESS host name field. When this field is set to an asterisk, multihome support allows the high availability manager to open and receive connections on all IP addresses available for the host.

With the introduction of Version 6.0.1 Fix Pack 2, processes can be classified into three separate categories:

Type 1

Processes that can operate only in single-IP mode. This category includes processes on Version 6.0 nodes or Version 6.0.1 nodes that do not have Version 6.0.1 Fix Pack 2 installed.

Type 2

Processes on Version 6.0.1 nodes that have Version 6.0.1 Fix Pack 2 installed, but do not have the DCS_UNICAST_ADDRESS host name field for the process set to an asterisk.

Type 3

Processes on Version 6.0.1 nodes that have Version 6.0.1 Fix Pack 2 installed and have the DCS_UNICAST_ADDRESS host name field for the process set to an asterisk.

The following interoperability rules apply to these process types:

- Type 1 and type 2 processes can interoperate.
- Type 2 and type 3 processes can also interoperate.
- Type 1 and type 3 processes cannot interoperate.

  Therefore, careful planning is necessary before converting processes to type 3 processes.

To enable interoperating with Version 6.0.1.2 processes:

Procedure

1. Add multihome capability by installing Version 6.0.1 Fix Pack 2. This step changes the existing type 1 processes to type 2 processes.
   a. Apply Version 6.0.1 Fix Pack 2 to all existing nodes. Nodes on which Version 6.0.1 Fix Pack 2 is installed continue to interoperate with nodes on which this fix pack is not installed, as long as the configuration does not change.
2. Stop all application servers.
3. Ensure that the deployment manager and all of the node agents are running.
4. Change the DCS_UNICAST_ADDRESS Host name field of each process to an asterisk. This step changes the existing type 2 processes to type 3 processes.
   a. In the administrative console, go to the configuration data for one of the processes on the new nodes:
      - For an application server, click **Servers** > **Server Types** > **WebSphere application servers** > *server_name*. Then, in the Additional Properties section, click > **Ports** > *port_name* .
      - For a node agent, click **System Administration** > **Node agents** > *node_name*. Then, in the Additional Properties section,, click > **Ports** > *port_name* .
      - For a deployment agent, click **System Administration** > **Deployment manager**. Then, in the Additional Properties section, click > **Ports** > *port_name* .

   b. Click **View associated transports** for the port that is associated with the DCS transport channel that you want to review.
   c. Click **DCS_UNICAST_ADDRESS**, and enter the name of the new host in the Host field.
   d. Click **OK**, and then click **Save**.
   e. Repeat the previous steps until the new host name is added for all of the processes on the new nodes.
   f. Select **Synchronize changes with nodes**, and then click **Save** again.
5. Stop all of the servers that contain the processes with configuration changes.
6. Restart these servers.

Results

All of the processes can communicate with each other.

What to do next

After multihome support is enabled, all of the new nodes that are added to the installation must have multihome support enabled. The base Version 6.0.1 code and Version 6.0.1 Fix Pack 2 must be installed before profiles are created and the node is added. If this procedure is not observed, manual configuration is required to enable the processes to connect.

Your configuration cannot contain a mix of type 1 and type 3 processes. To ensure a valid configuration:

1. Check for type 1 processes. Type 1 processes log the following message, which indicates that the host name for another process in the core group is an asterisk:

   ```
   HMGR0024W: An error was encountered while looking up the IP address for
   the host name of a core group member. The host name is * and the server
   name is myCell01\myCellManager01\dmgr. The member will be excluded from
   the core group.
   ```

2. Check for type 3 processes. Type 3 processes do not display in a view with type 1 processes, but can be detected by examining the HMGR0218 messages the various processes log. If the processes connect, the same message is logged across all processes. Specifically, processes that connect have the same view identifier and the same number of processes in the view.

   ```
   HMGR0218I: A new core group view has been installed. The core group is
   defaultCoreGroup. The view identifier is (8:0.spoletoCell01\
   spoletoCellManager01\dmgr). The number of members in the new view is 2.
   ```

Interoperating with Version 6.0.2 and later processes

The high availability manager supports multihomed hosts, which means that the product processes can communicate with each other even if they are running on different versions of the product. If you are running Version 6.0.2 and later processes that need to communicate with each other, you need to consider there are high availability manager-related issues.

Before you begin

Know the version levels of the processes that need to communicate with each other. If Version 6.0.1.2 processes exist, see “Interoperating with Version 6.0.1.2 processes” on page 77 for additional considerations.

About this task

When processes are created on Version 6.0.2 or later nodes, the host name field in the DCS_UNICAST_ADDRESS endpoint is set to an asterisk. When you use an asterisk in the host name field, multihome support allows the high availability manager to open and accept connections using any IP address that is valid for that machine.

Special considerations are required when Version 6.0.2 nodes interoperate with Version 6.0 and Version 6.0.1 nodes without Fix Pack 2 (6.0.1.2), because these earlier versions do not contain high availability manager multihome support. Version 6.0 and Version 6.0.1 processes do not recognize an asterisk as a valid value in the DCS_UNICAST_ADDRESS host name field. Therefore, these processes cannot connect to Version 6.0.2 processes that are configured with an asterisk in the DCS_UNICAST_ADDRESS host name field. When Version 6.0.2 processes must interoperate with Version 6.0 or Version 6.0.1 processes, the DCS_UNICAST_ADDRESS for all Version 6.0.2 processes must be configured to use a value other than an asterisk. This configuration is done by setting the host name field in the DCS_UNICAST_ADDRESS endpoint to a host name or to a textual IP address.

- If the host is configured to use a single IP address, a string host name, for example, myhost.mydomain, is sufficient.
- If the host is configured to use multiple IP addresses, then a textual IP address, for example, 192.168.0.2, is required.

As you add Version 6.0.2 nodes and create servers into a mixed-release cell, you must set the host name field of the DCS_UNICAST_ADDRESS for all processes on the new nodes, including the node agent, to one of the two values previously specified. You must then restart the Version 6.0.2 processes to pick up the new value.

To change the host name field for a process:

Procedure

1. In the administrative console, go to the configuration data for one of the processes on the new nodes:
   - For an application server, click **Servers > Server Types > WebSphere application servers >** *server_name*. Then in the Additional Properties section, click **> Ports >** *port_name* .
   - For a node agent, click **System Administration > Node agents >** *node_name*. Then in the Additional Properties section, click **> Ports >** *port_name* .
   - For a deployment agent, click **System Administration > Deployment manager**. Then in the Additional Properties section, click **> Ports >** *port_name* .
2. Click **View associated transports** for the port that is associated with the DCS transport channel you want to review.
3. Click **DCS_UNICAST_ADDRESS**, and enter the name of the new host in the Host field.
4. Click **OK**, and then click **Save**.
5. Repeat the previous steps until the new host name is added for all of the processes on the new nodes.
6. Select **Synchronize changes with nodes**, and then click **Save** again.
7. Stop all of the servers that contain the processes with configuration changes.
8. Restart these servers.

Results

All of the processes can communicate with each other.

What to do next

If Version 6.0.2 processes are not configured properly, the Version 6.0 and Version 6.0.1 processes might not start or might not be able to connect to the Version 6.0.2 processes.

The condition can be detected in one of the following ways:

- Version 6.0 and Version 6.0.1 processes log the following message:

  ```
  HMGR0024W: An error was encountered while looking up the IP address for
  the host name of a core group member. The host name is * and the server
  name is myCell01\myCellManager01\dmgr. The member is excluded from the
  core group.
  ```

  This message indicates that the host name for another process in the core group is an asterisk.
- The Version 6.0, Version 6.0.1, and Version 6.0.2 processes form separate views if the Version 6.0 and Version 6.0.1 processes do not connect to Version 6.0.2 processes. To detect these views, examine the HMGR0218 messages that the various processes logged. If the processes connect, the same message is logged across all processes. Specifically, processes that are connected have the same view identifier and the same number of processes in the view.

  ```
  HMGR0218I: A new core group view is installed. The core group is
  defaultCoreGroup. The view identifier is (8:0.spoletoCell01\
  spoletoCellManager01\dmgr). The number of members in the new view is 2.
  ```

Core group transports

Core group members communicate with each other over a specialized and dedicated network transport. Multiple transport implementations are supported, each with its own set of advantages and disadvantages.

You can use one of the following types of transports to set up communications between core group members. However, if possible, use a channel framework transport, because support for unicast transports is deprecated.

- Channel framework transport, which is the default transport
- Unicast transport

All of these transport options require TCP/IP connections for communication between core group members. The high availability discovery protocol ensures that these connections are opened, but the selected transport opens the connections. The discovery protocol also ensures, for each core group member, that connections are opened to all other members of that core group; thereby, ensuring that all running core group members are fully connected through TCP/IP.

Each core group member has a configured endpoint known as the DCS_UNICAST_ADDRESS. This endpoint contains the host and port information that indicates where the core group member is listening for TCP/IP connections.

Channel framework transport

The channel framework transport is the most flexible and scalable of the transport options and is the best option for most topologies. The channel framework transport provides the most security options for core group connections. However, the advanced features of the channel framework transport come at the cost of slightly lower performance. This performance impact is a concern in the most demanding topologies, where replication throughput is a primary objective.

The channel framework transport is the default transport type for core group member connections. If you use a channel framework transport, all communication occurs directly over the core group TCP/IP connections. The work of opening connections and sending or receiving data is delegated to the channel framework transport and the associated channel implementations.

For example, to use a channel framework transport for distribution and consistency services (DCS) messages, you must configure either a DCS transport chain, which is the default, or a DCS_SECURE transport chain, in addition to the DCS_UNICAST_ADDRESS endpoint. A DCS transport chain uses a TCP channel for network communication. A DCS_SECURE transport chain includes both a TCP channel and a secure sockets layer (SSL) channel to add support for encrypted communication.

The channel framework function provides a common model for connection management, and contains support for implementing various channels that you can combine into a transport chain. The ability to combine various types of channels makes it possible to create custom transport chains. The channel framework function also supports port sharing, which provides additional flexibility, and facilitates future customization.

If you use a channel framework transport, two different mechanisms are available to secure the core group network connections:

1. A lightweight third-party authentication (LTPA) token is used to authenticate all incoming connection requests if administrative security is enabled for the product.
2. SSL is used to encrypt all communications if the SSL version of a transport chain is selected.

You can use these mechanisms separately or combine them for the highest level of security possible.

Unicast transport

depfeat: Support for unicast transports is deprecated. You should use the channel framework transport as your transport type whenever it is possible to do so.

The communication mechanism of the unicast transport is similar to that of the channel framework transport. All communications occur over core group TCP/IP connections. The major difference is that a standard network connection, instead of a transport chain, is established and used to perform the communication between core group members.

Because a unicast transport does not go through the channel framework, this transport is somewhat faster than the channel framework transport. This performance improvement might be useful in topologies that make extensive use of memory-to-memory replication, and where the throughput that is obtained using a channel framework transport is not adequate.

Internal benchmarks show a gain in throughput using a unicast transport rather than a channel framework transport. However, this performance gain is achieved at the cost of using additional ephemeral ports.

A unicast transport has fewer security options than the channel framework transport. As with the channel framework transport, if administrative security is enabled for the product, an LTPA token is used to authenticate all incoming connection requests. However, no SSL encryption option is available for a unicast transport.

A unicast transport provides optimum performance with basic security. You trade the flexibility that a channel framework transport provides for improved performance. You are no longer able to do things like configure for SSL encrypted communication, or create a custom transport chain. However, in a typical application server environment, the unicast transport provides optimal performance for functions like memory-to-memory session replication.

Because a unicast transport uses more ephemeral ports than a channel framework transport, it might not scale as well as a channel framework transport does as the core group size increases.

Configuring core group IP caching

The caching of name-to-IP address information can be performed at many levels within the network communication software stack. These levels include within the operating system, the Java virtual machine, and within the product components, such as core groups. Core groups use caching to reduce the overhead that is associated with IP address name lookup. You can adjust the interval at which a core group IP cache is cleared.

Before you begin

By default, a core group cache is cleared every 60 minutes. You can determine the correct time interval for your environment.

About this task

You want to change the length of time that IP addresses are retained in a core group cache.

To change how frequently a core group cache is cleared:

Procedure

1. In the administrative console, click **Servers > Core Groups > Core group settings >** *core_group_name*.
2. In the Additional Properties section, click **Custom properties**.
3. Change the value that is specified for the IBM_CS_IP_REFRESH_MINUTES custom property. A core group cache cannot be cleared more frequently then once a minute.

   If the IBM_CS_IP_REFRESH_MINUTES property already exists, click on the property name and in the Value field, specify the length of time, in minutes, you want to wait before a core group cache is cleared.

   If this property does not already exist, click **New** and create it:

   a. In the Name field, specify `IBM_CS_IP_REFRESH_MINUTES`.
   b. In the Value field specify the length of time, in minutes, that you want to wait before a core group cache is cleared.

4. Click **OK** and then click **Review**.
5. Select **Synchronize changes with nodes**, and then click **Save**.
6. Restart all members of the core group.

Results

After the servers restart, all of the core group members run with the new discovery protocol setting.

Configuring core group socket buffers

Most operating systems provide program interfaces for performing operations involving the sending and receiving of data over sockets. Most operating systems also provide administrative capabilities to control the amount of memory allocated per socket that is used as data buffers.

Before you begin

- Use the performance monitoring infrastructure for the product to determine the average message size that the core group transport handles. If your operating system setting for the default buffer size is smaller than the average message size, make one of the following changes:
  - Change the default buffer size setting for your operating system. However, this action might be inappropriate because it might affect the operation of other applications running on this operating system.
  - Change the size of the socket buffer that the core group transport obtains. The value that is specified for the IBM_CS_SOCKET_BUFFER_SIZE core group custom property determines the size of the socket buffer that the core group transport obtains.

Table 11. Socket Buffer type, values for the property, and memory allocation. The following table shows the relationship between the values that can be specified for this property and the underlying memory allocation size per socket buffer type:

Socket Buffer Type	Property set to 0	Property set to 1	Property set to 2	Property set to 3
Unicast receive	Operating system default buffer size is used.	Buffer size is 64 kilobytes	Buffer size is 256 kilobytes	Buffer size is 1 megabyte
Unicast send	Operating system default buffer size is used.	Operating system default buffer size is used.	Buffer size is 64 kilobytes	Buffer size is 128 kilobytes
Multicast receive	Operating system default buffer size is used.	Buffer size is 512 kilobytes	Buffer size is 1 megabyte	Buffer size is 3 megabytes

About this task

You might want to change the size of your core group buffers in the following circumstances:

- You are directed to do so by IBM Support
- You are directed to do so during the course of installing another WebSphere product.
- You want to change the behavior of the core group transport without affecting the behavior of other sockets.
- You are trying to tune the network communication path of your system to your application.

To change the socket buffer space that the core group transport allocates:

Procedure

1. In the administrative console, click **Servers > Core Groups > Core group settings** *core_group_name*.

2. Under Additional Properties, click **Custom properties**.
3. Change the value that is specified for the IBM_CS_SOCKET_BUFFER_SIZE custom property.

   If the IBM_CS_SOCKET_BUFFER_SIZE property already exists, click the property name and specify either `0`, `1`, `2`, or `3`.

   If this property does not already exist, click **New** and create it:
   a. In the Name field, specify `IBM_CS_SOCKET_BUFFER_SIZE`.
   b. In the Value field specify one of the following strings:
      - `0`
      - `1`
      - `2`
      - `3`
4. Click **OK** and then click **Review**.
5. Select **Synchronize changes with nodes**, and then click **Save**.
6. Restart all members of the core group.

Results

After the servers restart, the core group members all run with the new socket buffer size settings.

Setting up IP addresses for high availability manager communications

There are situations where you must select a preferred IP address, or a range of IP addresses that you want the high availability manager to use for communication within a core group.

Before you begin

Determine the transport protocol the core group associated with the high availability manager uses for communications.

About this task

depfeat: Support for unicast transports is deprecated. If the core group is configured to use unicast transports, you should change the configuration settings to channel framework transports wherever it is possible to do so.

You should only set a preferred IP address if you want to restrict the high availability manager communications to a specific IP address. To configure a preferred IP address, complete the following steps:

Procedure

1. Make a list of all of the product processes that are on this machine.

   Include both application server processes and administrative processes, such as node agents or the deployment manager, in this list. If a preferred IP is set for one process on a machine, it should be set for all processes on that machine
2. Determine the textual form of the preferred IP address. For example, 9.5.87.124 or 10.1.2.2.
3. Update the IP address specified for the DCS unicast address for all processes identified in the first substep with the textual form of the preferred IP address.
   a. In the administrative console, navigate to the settings page for either an application server, a node agent, or a deployment manager.

      For an application server process click **Servers > Server Types > WebSphere application servers**.

      For a node agent process click **System Administration > Node agents**.

For a deployment manager process click **System Administration > Deployment manager**.

b. Select the appropriate process.

c. In the Additional Properties section, , click **Ports** to bring up the list of ports for the selected process. Then click on the port named **DCS_UNICAST_ADDRESS**.

d. Enter the preferred IP address in the Host field.

Following are the allowed values for the Host field:

- * (asterisk), which allows high availability manager communications on all NICs. This is the default value, but cannot be used if your processes must interact with V6.0 or V6.0.1 processes.
- Text IP address, such as 10.1.1.2, which restricts high availability manager communications to the specified NIC. Text IP addresses can be used if the host is configured to use a single IP address.
- A string host name of the form *myhost.mydomain*.

4. Click **Apply**, and then click **Save** to save your changes.

What to do next

The high availability manager uses the specified IP address for communication within the core group.

Specifying a preferred server for messaging requests

If a core group includes a cluster of application servers, and a messaging engine is configured for that cluster, any of the servers in that cluster can handle work items for the messaging engine. The default message provider that is included in the product is based on Service Integration Bus (SIB) technology, and is governed by the Default SIBus policy, which is a One of N policy. This policy ensures that only one of the application servers in the cluster is active at a time). You can modify the high availability group policy to specify that a specific cluster member handles the messaging work.

Before you begin

Before specifying a preferred server for messaging requests:

- You should review the following topics:
  - “High availability groups” on page 28
  - “High availability group policy modification guidelines” on page 35
  - “High availability group policies” on page 29
- You must determine:
  - The name of the core group that includes the server that you want to handle messaging requests.
  - The name of the high availability group for the messaging function.
  - The name of the policy that is associated with this high availability group
- You must create a new policy specific to the high availability group that controls the messaging engine cluster, if one does not already exist.

  It is possible for a single policy to govern several different high availability groups. Therefore, to modify the policy for cluster scoped control, you must create a new policy specific to the high availability group that controls the messaging engine cluster. See Chapter 6, “Creating a policy for a high availability group,” on page 45 for more information on how to create this policy.

  After you create the new policy and associate the policy with the high availability group for a given cluster, You can specify a preferred server for messaging requestws..

About this task

For high availability, you must configure a messaging engine to run in a cluster. However, you might want a specific cluster member to handle the messaging requests. Another member of the cluster should handle the messaging requests only if the preferred member fails.

Procedure

1. In the administrative console, click **Servers** > **Core Groups** > **Core group settings** > *core_group_name*.
2. Click the **Runtime** tab to determine both the name of the high availability group, and the name of the policy that is currently controlling the group. See Chapter 5, "Viewing high availability group information," on page 39 for more information on how to perform this step. You must have at least one of the group members running.
3. In the administrative console, click **Servers** > **Core Groups** > **Core group settings** > *core_group_name* > **Policies**.
4. Click the name of the policy that you want to modify.
5. Under Additional Properties, select **Preferred servers** and select the preferred servers for this policy.

   Click **Add** to move core group servers into the list of preferred servers.

   Select specific servers in the list and click **Move up** and **Move down** to adjust the order of the servers within the list. Make sure that the most preferred server is at the beginning of the list and the least preferred server is at the end of the list.
6. After you complete the preferred servers list, click **OK**.
7. Click **OK** and then click **Review**.
8. Select **Synchronize changes with nodes**, and then click **Save**.

Results

All work items for the messaging engine on the associated cluster are routed to the new preferred server.

Chapter 10. Viewing the core groups in a cell

A core group is a component of the high availability manager. A default core group, called DefaultCoreGroup, is created for each cell. A core group can contain application servers, proxy servers, node agents, and the deployment manager. A core group must contain at least one node agent or the deployment manager.

About this task

If you need to move servers between core groups, in preparation for the move, you can view the list of the different core groups contained in a cell to help determine to which core group you want to move a server.

To view the core groups that are in a cell:

Procedure

In the administrative console, click **Servers > Core Groups > Core group settings**.

Results

A list of the core groups that are in the cell is displayed.

What to do next

Click the name of one of the core groups to obtain specific information about that core group.

Core group collection

Use this page to view the core groups that are defined for your system. A core group is a component of the high availability manager function. A default core group, called DefaultCoreGroup, is created for each cell in the product environment. A core group can contain stand-alone servers, cluster members, node agents and the deployment manager. A core group must contain at least one node agent or the deployment manager.

To view this administrative console page, click **Servers > Core Groups > Core group settings** .

Click **New** to define a new core group. After a core group is defined, several fields become read-only. To change those fields, delete and redefine the core group.

To delete a core group, select the name of that core group and then click **Delete**. A core group must be empty before it can be deleted.

Name

Specifies the name of the core group. Click the core group name to edit the settings for that core group. This field is read-only.

Description

Specifies a description of the core group. This field is read-only.

Connected core groups

Specifies the core groups that are connected to this core group by access points. This field is read-only.

Chapter 11. Viewing core group members

A core group member is an application server, proxy server, the deployment manager, or a node agent that is a member of a high availability core group.

About this task

If you need to move servers between core groups, in preparation for the move, you can view the list of servers that belong to each core group to help determine which servers you want to move to a different core group.

To view the members of a core group:

Procedure

1. In the administrative console, click **Servers > Core Groups > Core group settings**. A list of core groups that are in the cell is displayed.
2. Click the name of one of the core groups and then click **Core group servers**.

Results

A list of the core group members is displayed.

What to do next

In the administrative console, you can click **Servers > Core Groups > Core group settings > Preferred coordinator servers** to determine which of the core group members are on the list of preferred coordinator servers for this core group. You should remove any servers that you intend to move to a different core group from this list before changing their core group association.

Core group servers collection

Use this page to view a list of the servers that are part of a core group. A core group server can be an application server, a deployment manager, or a node agent that is a member of a high availability core group. Use this page to move servers into a different core group. All members of a cluster must be in the same core group. If you select one or more members of a cluster, all of the members of that cluster must be moved.

To view this administrative console page, click **Servers > Core Groups > Core group settings**, click the name of an existing core group, and then click **Core group servers**.

To move one or more servers to another core group, select the check box next to the names of the servers that you want to move and click **Move**.

gotcha: You must stop a core group server before you move it to another core group.

Name

Specifies the names of the servers in the core group. This field is read-only. Click this field to specify custom properties for this server.

Node

Specifies the node that contains the core group server. This field is read-only.

Type

Specifies the server process type, which can be either deployment manager, node agent, or application server. Stand-alone application servers in the cell, managed nodes, and cluster members all display as application server types. This field is read-only.

Cluster name

Specifies the cluster name if the core group server is part of a cluster. If the core group server does not belong to a cluster, this field is blank. This field is read-only.

Core group server settings

Use this page to view the servers that are part of a core group and to define or edit custom properties for a core group server.

To view this administrative console page, click **Servers > Core Groups > Core group settings >** *core_group_name* **> Core group servers >** *server_name*.

Click **Custom properties** to define or edit a custom property for the core group server.

Node

Specifies the name of the node that contains the core group server. This field is read-only.

Name

Specifies the name of the core group server. This field is read-only.

Chapter 12. Moving core group members

When moving members to a different core group, remember that: each process can only be a member of one core group, and that all members of a given cluster must belong to the same core group.

Before you begin

- Review the topic, *Core groups (high availability domains)*.
- Determine which core group members you want to move, and to which core group you want to move them
- If you move one cluster member, all of the other members of the cluster are automatically relocated to the new core group.

About this task

You might need to move one or more core group members:

- To populate a newly created core group.
- To rebalance existing core groups.

The steps you perform to move either an application server, a deployment manager, or a node agent are slightly different. Depending on whether you are moving an application server, a deployment manager, or a node agent, complete one of the following steps.

gotcha: In general, you should not move a deployment manager.

Procedure

- Move one or more application servers to another core group.
  1. Stop the application servers that you want to move.
  2. In the administrative console, click **Servers** > **Core Groups** > **Core group settings** to display a list of the core groups in the topology.
  3. Click the name of the core group that contains the application servers that you want to move.
  4. On the configuration panel for this core group, under Additional Properties, click **Core group servers** to display the list of members of this core group. This list includes all of the application servers, node agents, and deployment managers that are members of this core group.
  5. In the Select column, select the application servers that you want to move to a new core group.

     You can only designate one target core group to which to move the selected application servers. If you need to move some of the application servers on this list to two or more other core groups, you have to repeat this step and the following step for each core group.
  6. Click **Move**. The **Core Groups** > **Core group settings** > ***core_group_name*** > **Core group servers** > **Move** administrative console panel is displayed. This panel lists the application servers that you selected to move and the core group to which they currently belong.
  7. Select the core group to which you want these application servers moved. The pull-down, under To core group, lists the core groups that are available on your system.
  8. Click **Apply**, and then click **Save**.
  9. Click **System Administration** > **Node agents**, select all running nodes, and then click **Synchronize** to synchronize your configuration changes to all of the running nodes.
  10. Restart the application servers that you moved.
- Move one or more node agents to another core group.
  1. Stop the node agents that you want to move.
  2. In the administrative console, click **Servers** > **Core Groups** to display a list of the core groups in the topology.

3. Click the name of the core group that contains the node agents that you want to move.
4. On the configuration panel for this core group, under Additional Properties, click **Core group servers** to display the list of members of this core group. This list includes all of the application servers, node agents, and deployment managers that are members of this core group.
5. In the Select column, select the node agents that you want to move to a new core group.

   You can only designate one target core group to which to move the selected node agents. If you need to move some of the node agents on this list to two or more other core groups, you have to repeat this step and the following step for each core group.
6. Click **Move**. The **Core Groups** > ***core_group_name*** > **Core group servers** > **Move** panel displays. This panel lists the node agents that you selected to move and the core group to which they currently belong.
7. Select the core group to which you want these node agents moved. The pull-down, under To core group, lists the core groups that are available on your system.
8. Click **Apply**, and then click **Save**.
9. Issue the `syncNode` command from the profile_root/`node_agent_profile/bin` directory to manually synchronize the updated configuration to the node.

   gotcha: You must perform this synchronization before you restart the moved node agent.
10. Restart the node agent that you moved.

- Move the deployment manager to another core group.
  1. Stop the deployment manager.
  2. Start a local mode scripting session under the deployment manager profile. Use the following command from the *profile_root*/`deployment_manager_profile/bin` directory:

```
wsadmin –conntype NONE –lang jython
```

  3. In the local mode scripting session, move the deployment manager. Use the following command:

```
AdminTask.moveServerToCoreGroup("-source <source_Core_Group>
   -target <target_Core_Group>
   -nodeName <node_name> -serverName <server_name>")
```

  4. Save the configuration changes. Use the following command:

```
AdminConfig.save()
```

  5. Restart the deployment manager.
  6. Click **System Administration** > **Node agents**, select all running nodes, and then click **Synchronize** to synchronize your configuration changes to all of the running nodes.

Results

After all of the restarts complete, all moved application servers, node agents, and deployment managers should belong to their new core group.

What to do next

- You can verify that the servers are in the correct core groups. For each core group, in the administrative console, click **Servers** > **Core Groups** > ***core_group_name*** > **Core group servers**, and look at the list of core group members that displays.
- You can set up core group bridges if any of the core groups need to communicate with each other. See the *Core group communications using the core group bridge service* topic for more information.

Core group server move options

Use this page to move one or more core group servers to a different core group. You must stop a core group server before you move it.

Servers can be moved from one core group to another, as long as the following core group requirements are not violated:

- A non-empty core group retains at least one node agent or deployment manager as a member of that group. (The high availability manager configuration change listeners are only available on the node agent or deployment manager servers.
- All members of a cluster must be members of the same core group. If one or more of the servers you are moving belongs to a cluster, you must move all of the members of that cluster. (A core group can span multiple product clusters.)

To view this administrative console page, click **Servers** > **Core Groups** > **Core group settings** > *core_group_name* > **Core group servers**. Select the servers to be moved and then click **Move**.

Extended information about the core group fields:

Move selected servers
Specifies the servers that you selected to be moved. It cannot be edited.

From core group
Specifies the name of the core group that you are moving the servers from. It can not be edited.

To core group
Specifies the core group to which these servers will belong.

Click **Apply** to make your changes effective. Click **Save** and save and synchronize your changes with all managed nodes.

Chapter 13. Configuring the core group bridge service

The core group bridge service can be configured to establish communication between core groups. A core group is a statically defined component of the high availability manager. To configure communication between core groups, use an access point group. An access point group is a collection of core groups that communicate with each other.

Before you begin

Before you configure a core group bridge service, become familiar with the content of the following topics:

- *Core groups*, which describes the functionality of a core group.
- *Creating a new core group (high availability domain)*, which describes how to configure a core group.
- *Configuring communication between core groups that are in the same cell*, which describes how to configure communication between core groups that are in the same cell.
- *Creating advanced core group bridge configurations*, which describes how to configure two or more core groups, such that the members of each core group can communicate with the members of the other core groups, even if one or more of the other core groups resides outside of a firewall.

About this task

You must configure the core group bridge service whenever two or more core groups are configured in the same cell. You must also configure the core group bridge if you need to share traffic among core groups that are in different cells. Configure the core group bridge to communicate between cells only when the service is required by another product component. By configuring the core group bridge service, the availability status of the servers in each core group is shared among all the configured core groups. You can configure core groups to communicate in the following ways:

Procedure

- Configure the core group bridge service for communication between two core groups that are in the same cell. You can configure each core group to communicate with one or more of the other core groups in the cell.
- Configure the core group bridge service for communication between two core groups that are in different cells. You can configure the core group in one cell to communicate with a core group in the other cell.
- Configure the core group bridge service for communication within the cell, and outside of the cell.
- Configure the core group bridge service for communication using a tunnel peer access point. This is the type of communication that you must configure if you are using a DMZ Secure Proxy Server for IBM WebSphere Application Server.

Results

Multiple core groups can communicate with each other.

What to do next

Continue configuring the high availability environment.

Core group communications using the core group bridge service

The core group bridge service can be configured to share availability information about internal product components between core groups. For example, by configuring the core group bridge service, each core group can be aware of the status of all of the application servers that are configured in all of the core groups. Use access point groups to define the core groups that communicate. Do not use the core group bridge service to share application information among core groups.

A core group is a statically defined component of the high availability manager. Each cell must have at least one core group. The product automatically creates a default core group called **DefaultCoreGroup** for each cell. If you create additional core groups, you can configure core group bridges and access point groups to enable the different core groups to communicate with each other, and share workload management information. If you do not configure a core group bridge between the core groups, the core groups are isolated from each other.

A core group bridge can be configured to enable the following communication scenarios:

- Communication between core groups within the same cell.
- Communication between core groups in different cells
- Communication between core groups within the same cell and one or more core groups in a different cell
- Communication between core groups across different networks
- Communication between core groups using a proxy peer access point.
- Communication between core groups using a tunnel peer access point. This type of communication must be configured if you are using a DMZ Secure Proxy Server for IBM WebSphere Application Server

Core group bridge overview

To configure communication between core groups, you must configure an *access point group*. An access point group is a collection of the core groups that communicate with each other. Add a *core group access point* to the access point group for each core group that needs to communicate.

A *core group access point* is a collection of server, node, and transport channel chain combinations that communicate for the core group. Each core group has one or more defined core group access points. The **DefaultCoreGroup** has one default core group access point. However, you might consider configuring more than one core group access point for a core group if that particular core group needs to be connected to other core groups that are on different networks.

The node, server, and transport channel chain combinations that are in a core group access point are called *bridge interfaces*. A server that hosts the bridge interfaces is a *core group bridge server*. The transport channel chain defines the set of channels that are used to communicate with other core group bridge servers. Each transport channel chain has a configured port that the core group bridge server uses to listen for messages from other core group bridge servers.

Each core group access point must have at least one core group bridge server. The core group bridge server provides the bridge interface for each core group access point. Because core group bridge servers within a core group access point serve as backups for each other, it is recommended that you have two core group bridge servers within each core group access point. Then, if one core group bridge server fails, the other core group bridge server can take over the failed core group bridge server's responsibilities.

If you are configuring communication between core groups that are in the same cell, create one access point group and add a core group access point for each core group that needs to communicate.

If you are configuring the core group bridge between core groups that are in different cells, you still use an access point group. However, you must create and configure the access point group for each cell. Each cell has an access point group that contains a core group access point for the core group that is in the cell, and a *peer access point* for each peer cell.

A peer access point references a core group access point that is configured in a different cell. Each access point group must have one peer access point for each different cell. Do not configure multiple peer access points that reference the same cell.

Each peer access point has one or more *peer ports* or one *proxy peer access point*.

A peer port corresponds to a bridge interface that is defined in the peer cell. You can define several peer ports for each peer access point.

Define a proxy peer access point if the peer access point cannot be reached directly by using a peer port, but can be reached by using another peer access point. The proxy peer access point specifies a peer access point that can communicate with the peer core group that cannot be reached directly. The proxy peer must have defined peer ports. Specify one proxy peer or one or more peer ports, but not both.

The following diagram shows a core group bridge configuration between two different cells that is using peer access points with peer ports.

Figure 1. Core group bridge configuration in two different cells

Communication within the cell and outside of the cell

The following example illustrates a configuration between three core groups that are in three different cells. Each cell has one access point group for communication between core groups in the cell. Each cell also has a defined *access_point_group_xyz* access point group, which contains one core group access point group for the core group that is in the cell, and one core group access point for each of the core groups in the other two cells.

Figure 2. Communication between core groups that are in the same cell with core groups outside of the cell

The following example shows the relationship between bridge interfaces and peer ports for the communication between the*cell_x* cell and the *cell_z* cell. In the *cell_x* cell, two bridge interfaces are defined. In the *cell_z* cell a peer access point exists for the *x_core_group_ap_2* core group access point with peer ports defined that correspond to the bridge interface information that is defined in the *cell_x* cell .

Figure 3. Bridge interfaces in one cell correspond to peer ports in the other cell

As a result, the *core_group_x* , *core_group_y* and *core_group_z* core groups can communicate with each other.

Communication between core groups across different networks

In this scenario, one core group is configured to communicate with two or more core groups in different cells across two or more networks. For example, a core group in the *cell_x* cell needs to communicate with core groups in the *cell_y* and *cell_z* cells. Create two access point groups in the *cell_x* cell. The*access_point_group_xy* access point group, in the *cell_x* cell contains a core group access point and a peer access point for the core group in the *cell_y* cell. The *access_point_group_xz* access point group in the *cell_x* cell contains a core group access point and a peer access point for the core group in the *cell_z* cell. The *cell_y* cell has an *access_point_group_xy* access point group, which has a core group access point and a peer access point for the *cell_x* cell. The *cell_z* cell has an *access_point_group_xz*access point group, which has a core group access point and a peer access point for the *cell_x* cell.

Figure 4. Core group communication across different networks

You can use a proxy peer access point when the core groups cannot directly communicate. The two core groups must have access to a single core group that can pass information between the two core groups. To understand what a proxy peer access point does, consider a connecting flight when flying on an airplane. To fly from Pittsburgh to London you first have to fly to New York City, where you change planes and then fly to London. New York City is the *proxy peer access point* for London.

When defining a proxy peer, the *x_core_group_2* core group in the *cell_x* cell cannot communicate directly with the core group in the *cell_z* cell. However, both core groups can communicate with the core group in the *cell_y* cell. To configure communication between the *cell_x* cell and the*cell_z* cell, you must configure two access point groups. The core group access point in the *cell_y* cell is in both the *access_point_group_xy* and *access_point_group_yz* access point groups. The following image shows an overview of a proxy peer configuration.

Figure 5. Core group communication using a proxy peer access point

Communication between core groups using a tunnel proxy peer access point

You can use a tunnel proxy peer access point when the core groups cannot directly communicate with each other because of a firewall. Typically, when a firewall exists between two core groups, the core groups cannot communicate with each other. However if a DMZ Secure Proxy Server for IBM WebSphere Application Server resides in the demilitarized zone (DMZ) outside of the firewall, you can create a tunnel access point group that enables you to establish a core group bridge tunnel between the core groups that are running on servers inside of the firewall, and the core groups that are running on the DMZ Secure Proxy Server for IBM WebSphere Application Server.

A tunnel access point group defines the access points that the core groups use to communicate with each other. The tunnel access point group consists of tunnel peer access points and core group access points. The tunnel peer access points are used to establish communication with the core groups that are running on the DMZ Secure Proxy Server for IBM WebSphere Application Server. The core group access points are used to establish communication between core groups that are running inside of the firewall.

Even though multiple core groups in a cell inside of the firewall need to communicate with a core group that are running in the DMZ Secure Proxy Server for IBM WebSphere Application Server, only one of the core groups can establish communication at a time. The DMZ Secure Proxy Server for IBM WebSphere Application Serverattempts to serially communicate with each core group inside of the firewall until it successfully establishes communication with one of the core groups. After communication is established, the DMZ Secure Proxy Server for IBM WebSphere Application Serverstay connected to that core group until communication fails.

The DMZ Secure Proxy Server for IBM WebSphere Application Server tries to reestablish communication with that core group twice. If the DMZ Secure Proxy Server for IBM WebSphere Application Server cannot reestablish communication with that core group, it tries to establish communication with one of the other core groups for which a tunnel peer access point exists in the tunnel access point group.

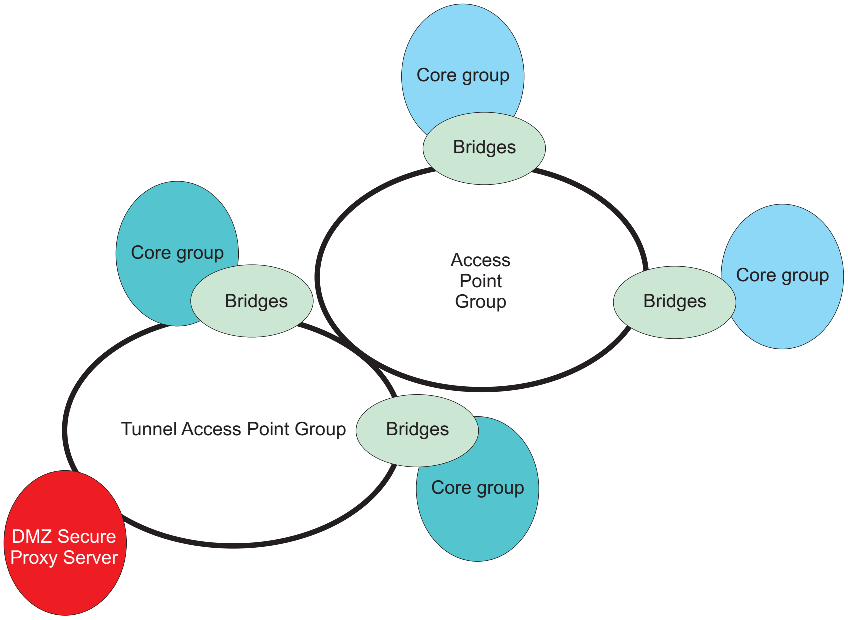


Figure 6. Core group communication using a tunnel peer access point

Configuring communication between core groups that are in the same cell

Use this task to configure communication between core groups that are in the same cell.

Before you begin

Configure two core groups with application servers that are in the same cell.

About this task

A cell, by default, contains a single core group, called DefaultCoreGroup. All processes in the cell are initially members of this core group. A single core group is usually sufficient. However, some topologies or special circumstances require multiple core groups. For example, the optimal number of servers in a core group is typically around 50. If the number of members in a core group exceeds this optimal number, you might need to create one or more additional core groups.

If you need to define multiple core groups within a cell, you should configure core group bridges between those core groups. If you do not configure these core group bridges, each core group is isolated from the other core groups in the cell.

After you configure the core group bridges between the core groups in a cell, each core group member can use the bridges to share its availability status with all of the other members of the configured core groups in that cell.

gotcha: When configuring core group bridges, remember the following requirements:

- Whenever a change is made in core group bridge configuration, including the addition of a new bridge, or the removal of an existing bridge, you must fully shutdown, and then restart all core group bridges in the affected access point groups.

- There must be at least one running core group bridge in each core group. If you configure two bridges in each core group, a single server failure does not disrupt the bridge functionality. Also, configuring two bridges enables you to periodically cycle out one of the bridges. If all the core group bridges in a core group are shutdown, the core group state from all foreign core groups is lost.

bprac: It is also recommended that:

- Core group bridges be configured in their own dedicated server process, and that these processes have their monitoring policy set for automatic restart.
- For each of your core groups, you set the IBM_CS_WIRE_FORMAT_VERSION core group custom property to the highest value that is supported on your environment.
- To conserve resources, do not create more than two core group bridge interfaces when you define a core group access point. You can use one interface for workload purposes and another interface for high availability. Ensure that these interfaces are on different nodes for high availability purposes. For more information, see the frequently asked question information on core group bridges.
- You should typically specify ONLY two bridge interfaces per core group. Having at least two bridge interfaces is necessary for high availability. Having more than two bridge interfaces adds unnecessary overhead in memory and CPU.

Complete the following actions to configure communication between core groups that are in the same cell.

Procedure

1. Configure an access point group to define the core groups that need to communicate.

   An access point group contains the core group access points for the core groups that need to communicate. Core group access points define the set of servers that provide access to the core group. To configure communication between core groups that are in the same cell, you can choose an existing access point group, such as the DefaultAccessPointGroup, which is created by default, or you can create a new access point group. To create a new access point group, complete the following steps:

   a. In the administrative console, click **Servers > Core Groups > Core group bridge settings > Access point groups > New**.

   b. Enter a name for the access point group that is unique within the cell.

   c. Add core group access points to your access point group.

      Choose any available core group access points for the core groups that need to communicate in the cell. A default core group access point is automatically created whenever a core group is created. Therefore, you should never have to create core group access points. The access point group that you create must have a core group access point for each core group in the cell that needs to communicate.

   bprac: When you configure communication between core groups in a single cell, you do not have to add any peer access points to an access point group Refer to the topic *Configuring the core group bridge between core groups that are in different cells* for more information about configuring communication between cells.

   If you use an existing access point group, choose an access point group that does not have peer access points. To configure an existing access point group, complete the following steps:

   a. In the administrative console, click **Servers > Core Groups > Core group bridge settings**. Your current configuration with any existing access point groups is displayed.

   b. Click **Access point groups >** *access_point_group_name*> **Core group access points**.

   c. Add core group access points to your access point group. Choose any available core group access points for the core groups that need to communicate.

2. Create bridge interfaces for each core group access point.

The bridge interfaces that you add provide access to the designated core group. Create at least one bridge interface for each core group access point. To ensure the availability of a core group access point, it is recommended that you configure two bridge interfaces for each access point.

gotcha: Even though it is possible to define multiple access points for a core group, you should only define a single access point to represent each core group

a. In the administrative console, click **Servers** > **Core Groups** > **Core group bridge settings** > **Access point groups** > *access_point_group_name* > **Core group access points**.

b. Click the name of a core group access point in the access point group, and then click **Show Detail**.

c. To create a new bridge interface, click **Bridge interfaces** > **New**.

d. Select a server to be the bridge interface.

bprac:

- A server that you select as a core group bridge interface should not be a server that handles production responsibilities, such as filtering requests to cluster members, or WebSphere proxy servers. Any server that is a core group bridge interface experiences extensive memory and CPU usage during core group bridge startup, and failover processing, which occurs when one of the bridges in a core group stops. If you cannot dedicate a server as a core group bridge interface, you should select a node agent as your core group bridge interface.
- You should typically specify ONLY two bridge interfaces per core group. Having at least two bridge interfaces is necessary for high availability. Having more than two bridge interfaces adds unnecessary overhead in memory and CPU.

e. Repeat these steps to create bridge interfaces for each core group access point in your access point group.

3. Optional: Configure the high availability manager protocol to establish transparent bridge failover support.

During core group bridge state rebuilds, cross-core group state can be moved between running bridges. This situation might cause the data to be temporarily unavailable until the bridge has completed the rebuild process.

If you are running on Version 7.0.0.1 or later, set the IBM_CS_HAM_PROTOCOL_VERSION core group custom property to `6.0.2.31` for all of your core groups to avoid a possible high availability state outage during core group bridge failover. When this custom property is set to `6.0.2.31`, the remaining bridges recover the high availability state of the failed bridge without the data being unavailable in the local core group.

Complete the following actions to set the IBM_CS_HAM_PROTOCOL_VERSION core group custom property to `6.0.2.31` for all of your core groups.

a. Shut down all core group bridges in all of your core groups.

b. Repeat the following actions for each core group in each of your cells:

1) In the administrative console, click **Servers** > **Core Groups** > **Core group settings** > *core_group_name* > **Custom properties**.

2) Specify `IBM_CS_HAM_PROTOCOL_VERSION` in the **Name** field, and `6.0.2.31` in the **Value** field.

3) Save your changes.

c. Synchronize your changes across the topology.

d. Restart all of the bridges in the topology.

All of the core groups within this topology are using the 6.0.2.31 high availability manager protocol.

Results

The core groups that are in the same cell and configured in an access point group can communicate.

Example

In the *cell_x* cell, there are the *x_core_group_1*, *x_core_group_2*, and *x_core_group_3* core groups. Each core group already has a core group access point. The following image illustrates an access point group between the core groups in the *cell_x* cell and an example of the configuration in the administrative console.

Figure 7. Three core group access points in the same cell belong to the same access point group.

Perform the following steps to configure communication between the three core groups in the *cell_x* cell:

1. Create the `x_access_point_group` access point group. Add a core group access point to the access point group for each core group that is in the cell. In this example, add the `x_core_group_ap_1` , `x_core_group_ap_2`, and `x_core_group_ap_3` access points to the `x_access_point_group` access point group.
2. Create bridge interfaces for each core group access point. The following diagram illustrates the bridge interfaces for the`x_core_group_ap_2` core group access point:

Figure 8. Core group access points contain one or more bridge interfaces.

By creating an access point group and adding all core groups in the cell to the access point group, you enabled communication between all the core groups that are in the *cell_x* cell.

What to do next

You can configure this cell to communicate with core groups in other cells.

Configuring core group bridge communication between cells that contain multiple core groups

You might need to configure core group bridge communication between cells that contain multiple core groups if one of the cells contains processes that need access to the high availability manager state information of processes that are running in a different cell. For example, a WebSphere Application Server proxy server or an On Demand Router might need to route requests to cluster members in a different cell and that other cell contains multiple core groups.

Before you begin

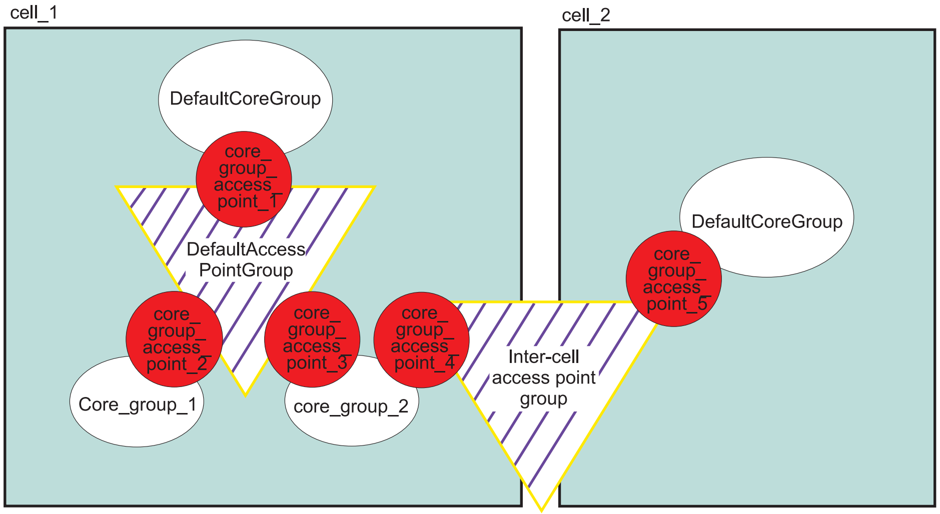
Verify that the following conditions exist:

- The cells between which you are configuring core group bridge communication have unique names. You cannot set up a core group bridge between two cells that have the same name.
- At least one of the cells contains multiple core groups, and core group bridge communication must already exist between the core groups that reside in that cell. The access point group that is used to establish this communication continues to handle communication between the core groups within the same cell even after communication is established between the two cells.

About this task

To enable core group bridge communication between cells that contain multiple core groups, you must create an inter-cell access point group to handle communication between one of the core groups in each of the cells. This inter-cell access point group, working in conjunction with the existing access point group, allows state information from within one of the cell to be propagated to the other cell.

The following figure illustrates this topology where DefaultAccessPointGroup is the access point group that enables communication between the three core groups in cell_1, and the inter-cell access point group enables communication between the core groups in cell_1 and the core group in cell_2.


cell_1
cell_2
DefaultCoreGroup
core_ group_ access_ point_1
DefaultAccess PointGroup
core_ group_ access_ point_2
core_ group_ access_ point_3
core_ group_ access_ point_4
Core_group_1
core_group_2
Inter-cell access point group
DefaultCoreGroup
core_ group_ access_ point_5

You can use core group bridge custom properties to set up advanced configurations for a core group bridge.

gotcha: When configuring core group bridges, remember the following requirements:

- Whenever a change is made in core group bridge configuration, including the addition of a new bridge, or the removal of an existing bridge, you must fully shutdown, and then restart all core group bridges in the affected access point groups.
- There must be at least one running core group bridge in each core group. If you configure two bridges in each core group, a single server failure does not disrupt the bridge functionality. Also, configuring two bridges enables you to periodically cycle out one of the bridges. If all the core group bridges in a core group are shutdown, the core group state from all foreign core groups is lost.

bprac: It is also recommended that:

- Core group bridges be configured in their own dedicated server process, and that these processes have their monitoring policy set for automatic restart.
- For each of your core groups, you set the IBM_CS_WIRE_FORMAT_VERSION core group custom property to the highest value that is supported on your environment.
- To conserve resources, do not create more than two core group bridge interfaces when you define a core group access point. You can use one interface for workload purposes and another interface for high availability. Ensure that these interfaces are on different nodes for high availability purposes. For more information, see the frequently asked question information on core group bridges.
- You should typically specify ONLY two bridge interfaces per core group. Having at least two bridge interfaces is necessary for high availability. Having more than two bridge interfaces adds unnecessary overhead in memory and CPU.

Complete the following procedure to configure core group bridge communication between cells that contain multiple core groups.

Procedure

1. Create a peer access point, and peer ports to specify the remote end points in the cell to which you want to establish communication from the multi-core group cell.

   Within each peer access point, configure a peer port that corresponds to each bridge interface in the other cell (cell_2 in the preceding figure). Before you add a peer access point, determine the following information about the other cell:

   - Cell name
   - Core group name
   - Core group access point name
   - Host and port information. The host and port correspond to the bridge interfaces that are configured in the other cell. You must specify a peer port for each bridge interface that is in the other cell.

   a. In the administrative console, click **Servers > Core Groups > Core group bridge settings > Access point groups >***access_point_group* **> Peer access points > New**.

      access_point_group is the name of the access point group that already exists in the multi-core group cell.

   b. Configure your peer access point.

      In addition to specifying a name for the peer access point, you must complete the following actions:

      - Specify the other cell where the peer access point resides.
      - Specify the name of the core group in the other cell to which the peer access point belongs.
      - Select **Use peer ports**.
      - Select the default level of access (**full access**) that you want this other cell to have to the multi-core group cell.

   c. Click **Next**, and then **Finish**.

   d. Save your configuration changes.

2. Create an inter-cell access point group in the multi-core group cell.

   a. In the administrative console, click **Servers > Core Groups > Core group bridge settings > Access point groups > New**.

   b. Click **Next**

   c. Select a core group from the **Available core group access points** list, and then click the right arrow to add the selected core group to the access point group.

      The selected core group should be the one that you want to use to exchange proxy state information between the core groups in the two cells. Remember that a peer access point can only represent one core group in another cell. Therefore, make sure you only add one core group from the multi-core group cell to the inter-cell access point group.

   d. Click **Next**

   e. Select the appropriate peer access point from the**Available peer access points** list, and then click the right arrow to add the selected peer access point to the access point group.

      The selected peer access point must be the one you previously created to communicate with the other cell.

   f. Click **Next**, and then click **Finish**.

   g. Save your configuration changes.

3. Create a peer access point, and peer ports to specify the remote end points in the multi-core group cell to which you want to establish communication from the other cell.

   Within each peer access point, configure a peer port that corresponds to each bridge interface in the multi-core group cell (cell_1 in the preceding figure). Before you add a peer access point, determine the following information about the multi-core group cell:

   - Cell name
   - Core group name

- Core group access point name
- Host and port information. The host and port correspond to the bridge interfaces that are configured in the other cell. You must specify a peer port for each bridge interface that is in the other cell.

a. In the administrative console, click **Servers > Core Groups > Core group bridge settings > Access point groups >***access_point_group* **> Peer access points > New**.

   access_point_group is the name of the access point group that already exists in the other cell.

b. Configure your peer access point.

   In addition to specifying a name for the peer access point, you must complete the following actions:

   - Specify the other cell where the peer access point resides.
   - Specify the name of the core group in the other cell to which the peer access point belongs.
   - Select **Use peer ports**.
   - Select the default level of access (**full access**) that you want the multi-core group cell to have to this other cell.

c. Click **Next**, and then **Finish**.

d. Save your configuration changes.

4. Create an inter-cell access point group in the other cell.

   a. In the administrative console, click **Servers > Core Groups > Core group bridge settings > Access point groups > New**.

   b. Click **Next**

   c. Select a core group from the **Available core group access points** list, and then click the right arrow to add the selected core group to the access point group.

      The selected core group should be the one that you want to use to exchange proxy state information between the core groups in the two cells. Remember that a peer access point can only represent one core group in another cell. Therefore, make sure you only add one core group from the other cell to the inter-cell access point group.

   d. Click **Next**

   e. Select the appropriate peer access point from the**Available peer access points** list, and then click the right arrow to add the selected peer access point to the access point group.

      The selected peer access point must be the one you previously created to communicate with the other cell.

   f. Click **Next**, and then click **Finish**.

   g. Save your configuration changes.

5. Configure bridge interfaces for your core group access point.

   Create bridge interfaces for the core group in the other cell if it is the only core group in that cell, which is the topology illustrated in the preceding figure. If there are multiple core groups in both cells, select any of the core groups in the other cell to use for inter-cell communication. If the core group you select already has bridge interfaces defined, skip this step.

   gotcha: When configuring core group bridges, remember the following requirements:

   - Whenever a change is made in core group bridge configuration, including the addition of a new bridge, or the removal of an existing bridge, you must fully shutdown, and then restart all core group bridges in the affected access point groups.
   - There must be at least one running core group bridge in each core group. If you configure two bridges in each core group, a single server failure does not disrupt the bridge functionality. Also, configuring two bridges enables you to periodically cycle out one of the bridges. If all the core group bridges in a core group are shutdown, the core group state from all foreign core groups is lost.

   a. In the administrative console, click **Servers > Core Groups > Core group bridge settings > Access point groups >***access_point_group* **> Core group access points**

   b. Select one of the listed core group access points. Then click **Show detail > Bridge interfaces > New**.
   c. Select a node, server, and transport chain for your bridge interface.
   d. Click **Apply**.
   e. Repeat this set of steps to add more bridge interfaces to the core group access point.

      Define two bridge interfaces for each core group access point to make your bridges highly available. When you define two core group bridge servers, if one of these two servers fails, the other server handles any pending communication, thereby preventing an interruption in the communication between the core groups. It is also recommended for the bridge interfaces of a core group to be on different nodes.

6. Optional: Configure the high availability manager protocol to establish transparent bridge failover support.

   When a core group bridge server restarts, state information, such as WLM data, and ODR or WebSphere Application Server proxy server routing data, is lost until the state information is recovered by a bridge in the same core group. The length of time the data is lost varies depending on the number of core groups and the amount of state data, but the length of time can be a minute or longer. During this time, ODRs and WebSphere Application Server proxy servers might report 503 errors, and JNDI look-ups for objects in remote core groups might fail.

   If you are running on Version 7.0.0.1 or later, you can avoid such outages, if you enable the transparent bridge failover protocol by setting the IBM_CS_HAM_PROTOCOL_VERSION custom property to 6.0.2.31 in each core group. When this custom property is set to `6.0.2.31`, the remaining bridges recover the high availability state of the failed bridge without the data being unavailable in the local core group.

   Complete the following actions to set the IBM_CS_HAM_PROTOCOL_VERSION core group custom property to `6.0.2.31` for all of your core groups.

   a. Shut down all core group bridges in all of your core groups.
   b. Repeat the following actions for each core group in each of your cells:
      1) In the administrative console, click **Servers > Core Groups > Core group settings >** *core_group_name* **> Custom properties**.
      2) Specify `IBM_CS_HAM_PROTOCOL_VERSION` in the **Name** field, and `6.0.2.31` in the **Value** field.
      3) Save your changes.
   c. Synchronize your changes across the topology.
   d. Restart all of the bridges in the topology.

   All of the core groups within this topology are using the 6.0.2.31 high availability manager protocol.

7. Save and synchronize changes, and restart all of the core group bridge servers in both cells to apply the changes.

What to do next

Continue configuring the high availability environment.

Configuring core group communication using a proxy peer access point

Use this task to configure communication between two core groups when they cannot communicate with each other directly through peer ports.

Before you begin

Configure a proxy peer access point to communicate with a core group if you cannot use peer ports. Use a core group that has configured a peer access point with peer ports to the core group that you want to communicate with. Before completing this task, make sure that you have access to a core group that can communicate with the core group that you cannot communicate with directly.

About this task

Use a proxy peer to communicate with a core group that you cannot access directly. This task describes how to configure a proxy peer with three core groups that are each in different cells. The *core_group_x* and *core_group_y* core groups can communicate directly with each other through peer ports. The *core_group_y* and *core_group_z* core groups can also communicate with each other though peer ports. However, the *core_group_x* core group cannot communicate with the *core_group_z* core group. To establish that communication, the *core_group_x* core group has a peer access point that is a proxy peer. The proxy peer is the peer access point to the *core_group_y* core group.

Complete the following actions to configure core group communication using a proxy peer access point.

Procedure

1. Configure the *core_group_x* and *core_group_y* core groups to communicate with each other by creating an access point group.
2. Configure the *core_group_y* and *core_group_z* core groups to communicate with each other by creating another access point group.

   When you do this step, create a second access point group in cell 2. The *core_group_2* core group communicates with both the *cell_x* and *cell_z* cells over two different networks.
3. Configure a peer access point that has a proxy peer. Create a new peer access point in the access point group that you created between the *core_group_x* and *core_group_y* core groups.
   a. In the administrative console, click **Servers > Core Groups > Core group bridge settings > Access point groups >** *access_point_group_name* **> Peer access points**. Create a new peer access point, or select an existing access point.
   b. Enter a unique name for the peer access point. For the other cell, core group, and core group access point values, use the properties of the*core_group_z* core group.
   c. Click **Use a proxy peer access point**. Select the proxy peer access point that is the peer access point that you created in the *cell_x* cell that refers to the core group access point in the *cell_y* cell.

Results

The *core_group_x* core group can communicate with the *core_group_z* core group by using a proxy peer.

Example

The following example shows the configurations in each cell when you configure communication between the *cell_x* and *cell_z* cells using a proxy peer access point:

Figure 9. Example proxy peer configuration

What to do next

By completing this task, you enabled one-way communication between the *core_group_x* and *core_group_z* core groups. If you want to configure communication both ways, you must repeat these steps, configuring a peer access point in the *core_group_z* core group that contains a proxy peer.

Configuring communication with a core group that resides on a DMZ Secure Proxy Server for IBM WebSphere Application Server

This task describes the steps that you must perform to establish communication between a cell inside of a firewall, and a DMZ Secure Proxy Server for IBM WebSphere Application Server outside of the firewall.

Before you begin

- Create a DMZ Secure Proxy Server for IBM WebSphere Application Server on your machine that is outside of the firewall, if one does not already exist.
- Configure core group bridges between your core groups that are in located inside of the firewall but reside in different cells, if they do not already exist.
- Read the topic *Advanced core group bridge configurations*, which describes how a tunnel access point group is used to set up a core group bridge tunnel between a cell inside of a fire wall, and a DMZ Secure Proxy Server for IBM WebSphere Application Server

About this task

gotcha: When configuring core group bridges, remember the following requirements:

- Whenever a change is made in core group bridge configuration, including the addition of a new bridge, or the removal of an existing bridge, you must fully shutdown, and then restart all core group bridges in the affected access point groups.
- There must be at least one running core group bridge in each core group. If you configure two bridges in each core group, a single server failure does not disrupt the bridge functionality. Also, configuring two bridges enables you to periodically cycle out one of the bridges. If all the core group bridges in a core group are shutdown, the core group state from all foreign core groups is lost.

bprac: It is also recommended that:

- Core group bridges be configured in their own dedicated server process, and that these processes have their monitoring policy set for automatic restart.
- For each of your core groups, you set the IBM_CS_WIRE_FORMAT_VERSION core group custom property to the highest value that is supported on your environment.
- To conserve resources, do not create more than two core group bridge interfaces when you define a core group access point. You can use one interface for workload purposes and another interface for high availability. Ensure that these interfaces are on different nodes for high availability purposes. For more information, see the frequently asked question information on core group bridges.
- You should typically specify ONLY two bridge interfaces per core group. Having at least two bridge interfaces is necessary for high availability. Having more than two bridge interfaces adds unnecessary overhead in memory and CPU.

Complete the following actions to create a tunnel access point group that contains the core group access point for the DMZ Secure Proxy Server for IBM WebSphere Application Server, and a tunnel peer access point that represents the cell that is located inside the firewall.

Procedure

1. In the administrative console, click **Servers > Core Groups > Core group bridge settings > Tunnel templates > New** to create a new tunnel template that will represent the core group bridge tunnel settings that can be exported to the DMZ Secure Proxy Server for IBM WebSphere Application Server.
2. Select the core group access points that you want to include in this group.

   When specifying the core group access points for the tunnel access point group, use the arrows to place the core group access points in the correct order. The specified order determines the order in which the DMZ Secure Proxy Server for IBM WebSphere Application Server defines the peer core groups of a tunnel peer access point. During startup, the proxy server attempts to connect to the peer core groups according to the order in which they are listed.

3. Click **OK**.
4. Click **Tunnel templates**, select the name of the template that you just created, and then click **Export**.

   The file is exported to the WAS_DMGR_PROFILE_ROOT/TUNNEL_TEMPLATE_NAME.props file.
5. On the DMZ Secure Proxy Server for IBM WebSphere Application Server, import the tunnel template settings into the DMZ Secure Proxy Server for IBM WebSphere Application Server configuration file.

   To import the tunnel template, issue one of the following commands:

   ```
   $AdminTask importTunnelTemplate -interactive
   ```

   or

   ```
   $AdminTask importTunnelTemplate {-inputFileName tunnel_template_name
        -bridgeInterfaceNodeName DMZ_PROXY_NODE_NAME
        -bridgeInterfaceServerName secure_proxy_name}
   ```

   and then issue the `$AdminConfig save` command.

   Where *tunnel_template_name* is the name that you gave the tunnel template that you just created, and *secure_proxy_name* is the name of your DMZ Secure Proxy Server for IBM WebSphere Application Server.
6. Optional: Configure the high availability manager protocol to establish transparent bridge failover support.

   During core group bridge state rebuilds, cross-core group state can be moved between running bridges. This situation might cause the data to be temporarily unavailable until the bridge has completed the rebuild process.

   If you are running on Version 7.0.0.1 or later, set the IBM_CS_HAM_PROTOCOL_VERSION core group custom property to `6.0.2.31` for all of your core groups to avoid a possible high availability state outage during core group bridge failover. When this custom property is set to `6.0.2.31`, the remaining bridges recover the high availability state of the failed bridge without the data being unavailable in the local core group.

   Complete the following actions to set the IBM_CS_HAM_PROTOCOL_VERSION core group custom property to `6.0.2.31` for all of your core groups.

   a. Shut down all core group bridges in all of your core groups.

   b. Repeat the following actions for each core group in each of your cells:

      1) In the administrative console, click **Servers** > **Core Groups** > **Core group settings** > *core_group_name* > **Custom properties**.

      2) Specify `IBM_CS_HAM_PROTOCOL_VERSION` in the **Name** field, and `6.0.2.31` in the **Value** field.

      3) Save your changes.

   c. Synchronize your changes across the topology.

   d. Restart all of the bridges in the topology.

   All of the core groups within this topology are using the 6.0.2.31 high availability manager protocol.

Results

A tunnel access point group is created that contains the core group access point for the DMZ Secure Proxy Server for IBM WebSphere Application Server, and a tunnel peer access point that represents the cell that is located inside the firewall.

Tunnel access point group collection

Use this page to view the tunnel access point groups that are defined for your core groups. The tunnel access point group includes core group access points and tunnel peer access points for enabling communication between a DMZ Secure Proxy Server for IBM WebSphere Application Server and a WebSphere Application Server, Network Deployment cell.

You can also use this page to create an additional tunnel access point group, or to delete an existing tunnel access point group.

To access this administrative console page, click **Servers > Core groups > Core group bridge settings > Tunnel access point groups**.

If you want to create an additional tunnel access point group, click **New**. Clicking new starts the Tunnel Access Point Group wizard. If you want to delete an existing tunnel peer access point group, select the tunnel access point group that you want to delete, and then click **Delete**.

Name

Specifies the name of a defined tunnel access point group.

Tunnel access point group settings

Use this page to modify the tunnel peer access points and the core group access points that belong to this access point group. A tunnel access point group defines the access points that a set of core groups use to communicate with each other, even though they reside on opposite sides of a firewall. Access points can be either tunnel peer access points or core group access points. The core group access points enable core groups in the same cell to communicate with each other. Tunnel peer access points enable core groups residing outside of the firewall to communicate with core groups residing inside of the firewall.

A tunnel access point group must include at least one core group access point to represent a core group in the local cell. If the access point group is located outside of a firewall, it must also contain at least one tunnel peer access point that represents a core group that is part of a cell that is located inside of the firewall.

To access this administrative console page, click **Servers > Core Groups > Core group bridge settings > Tunnel access point groups** *tunnel_access_point_group_name*.

From this page, you can edit the core group access points or the tunnel peer access points that belong to the selected tunnel access point group.

Name

Specifies the name of the tunnel access point group. The name of a tunnel access point group must be unique within a cell.

Member communication key

Specifies the name that core group members use to establish communication with other core group members that are defined as part of this tunnel access point group.

The default value is the name of the tunnel access point group.

Access points

Click **Core group access points** to define core group access points as part of this tunnel access point group. Click **Tunnel peer access points** to define tunnel peer access points as part of this tunnel access point group.

Tunnel peer access point collection

Use this page to view the tunnel peer access points that are defined for your core groups. Tunnel peer access points define the set of servers that provide access to core groups that reside in different cells, and one cell is located on a DMZ Secure Proxy Server for IBM WebSphere Application Server, while the other cell is located inside of the firewall. At least one tunnel peer access point must be defined for each cell, that is located on the DMZ Secure Proxy Server for IBM WebSphere Application Server, that needs to communicate with one or more of the core groups that are located inside of the firewall.

You can also use this page to create an additional tunnel peer access point, or delete an existing tunnel peer access point.

To access this administrative console page, click **Servers > Core Groups > Core group bridge settings > Tunnel peer access points**.

If you want to create a new tunnel peer access point, click **New**. If you want to delete an existing tunnel peer access point, select the tunnel peer access point that you want to delete, and then click **Delete**.

Name

Specifies the name of a defined tunnel peer access points that you can add to a tunnel access point group.

Tunnel peer access point settings

Use this page to configure a tunnel peer access point. A tunnel peer access point is used to establish communication between core groups that are in different cells, when one of the cells is located on a DMZ Secure Proxy Server for IBM WebSphere Application Server, and the other is located inside of the firewall. A tunnel peer access point corresponds to a core group access point in the peer cell. The tunnel peer access point communication settings are specified by using one or more peer endpoints or a proxy peer.

A tunnel peer access point must contain either peer ports or a proxy peer access point, but not both. When the tunnel peer access point is directly accessible within its tunnel access point group, specify peer ports. When the tunnel peer access point can be reached only indirectly, use a proxy tunnel peer access point. A proxy tunnel peer access point is used to identify the communication settings for the tunnel peer access point that cannot be accessed directly. The proxy tunnel peer access point specifies a peer access point that can communicate with the appropriate destination core group. The specified proxy tunnel peer access point must be a tunnel peer access point that has defined ports.

To view this administrative console page, click **Servers > Core Groups > Core group bridge settings > Tunnel peer access points** *tunnel_access_point_name*.

Name

Specifies the name of the tunnel peer access point. The name must be unique within the local cell.

Cell

Specifies the cell in which the tunnel peer access point resides.

gotcha: This property is case sensitive. The value you specify must exactly match the name of the cell in which the peer access point resides. For example, if WASCell05 is the name of the cell that contains the peer access point, you must specify `WASCell05` as the value for this property. If you specify `wascell05` as the value for this property, communication between the two core groups is not established.

Retry delay

Specifies, in seconds, the amount of time that you want the core group bridge service to wait before attempting to reconnect to a bridge. The default value is 30.

SSL configuration

Specifies whether to use SSL to establish a secure connection.

If SSL is selected, you must also select one of the following options:

- `Centrally managed`, if you want the product to manage the secure connections.

- `Specific to this endpoint`, if you want to specify a specific SSL configuration that is to be used to establish secure connections. When you select this option, you must also select the SSL configuration that you want used to establish secure connections.

Cell-level access

Specifies the level of access that a server from another cell is given to the local cell when that server uses this access point to establish communication with the local cell.

- **Full access** enables the communicating server to read data from and write data to the local cell. This level of access is appropriate if there is no reason to restrict read or write access to the local cell.
- **Read only** enables the communicating server to read data from the local cell, but prevents that server from writing data to the local cell. This level of access is appropriate if applications running in other core groups need to access data that is contained in the local cell but you want to make sure that the data stored on the local cell is not changed.
- **Write only** enables the communicating server to write data to the local cell, but prevents that server from reading data from the local cell. This level of access is appropriate if applications running in other core groups need to write data to the local cell, but the data stored on the local cell is sensitive. For example, the local cell might contain customer account numbers, and you do not want applications that resides outside of the local cell to read this information.

Tunnel peer access point selection

Use this page to control which tunnel peer access points are associated with this tunnel access point group. You can also use this page to create new tunnel peer access points for this tunnel access point group, or delete an existing tunnel peer access points.

To view this administrative console page, click **Servers > Core Groups > Core group bridge settings > Tunnel access point groups >** *tunnel_access_point_group_name* **> Tunnel peer access points**.

To see the configuration settings for a tunnel peer access point, select that tunnel peer access point, and then, click **Show details** for the list that contains that tunnel peer access point.

To create a new tunnel peer access point, click **New**.

To add an existing tunnel peer access point to this tunnel access point group, select one of the available tunnel peer access points, and click the first arrow to move that tunnel peer access point to the Tunnel peer access points list for this tunnel access point group.

To remove a tunnel peer access point from this tunnel access point group, select one of the tunnel peer access points from the list of tunnel peer access points that are contained in this access point group, and then click the second arrow to move the tunnel peer access point back to the Available tunnel peer access points list.

To delete a tunnel peer access point, select the tunnel peer access point that you want to delete, and click **Delete**.

When you finish adjusting your tunnel access points, click **OK** , and then click **Save** to save and synchronize your changes with all managed nodes.

Available tunnel peer access points

Specifies a list of defined tunnel peer access points that are available for you to add to this tunnel access point group.

Tunnel peer access points in *tunnel_access_point_group*

Specifies a list of tunnel peer access points that are in this tunnel access point group.

Tunnel templates settings

Use this page to edit the properties of a tunnel access point group template.

To change the properties of an existing template, in the administrative console, click **Servers > Core Groups > Core group bridge settings > Tunnel templates**.

Name

Specifies the name of the template.

Use SSL

Specifies whether SSL is used to establish secure connections for tunnel access point groups that are created using this template.

Tunnel access point group

Lists the name of the tunnel access point group whose core group access point settings are to be exported.

gotcha: The tunnel peer access points that belong to a tunnel access point group are not exported as a part of a tunnel template.

Create new tunnel access point group

Click **Create new tunnel access point group** to use this template to create a new tunnel access point group. Clicking this option starts the Create a new tunnel access point group wizard.

Tunnel templates collection

Use this page to view a list of the tunnel templates that are defined for a tunnel access point group. You can also use this page to create a new template, delete an existing template, or export relevant settings from the cell that is inside of the firewall to the cell that is outside of the firewall.

Exporting relevant settings simplifies the process of setting up your core group bridge configuration on a DMZ Secure Proxy Server for IBM WebSphere Application Server. The properties file for an exported tunnel template is saved in the ${USER_INSTALL_ROOT} directory that applies to the management profile for the DMZ Secure Proxy Server for IBM WebSphere Application Server.

To view the templates that are available for creating a new tunnel access point group, in the administrative console, click **Servers > Core Groups > Core group bridge settings > Tunnel templates**.

On this page, you can click **New** to create a new template, click **Delete** to delete a template, and click **Export** if you want to want to export relevant settings to a cell that resides outside of the firewall.

Name

Specifies the name of the tunnel access point group template.

Use SSL

Specifies whether this tunnel access group template is configured to use Secure Sockets Layer (SSL) to establish a secure connection between the core group members.

Tunnel Access Point Group

Specifies the name of a defined tunnel access point group on which this template is based.

Peer core group collection

Use this page to view the peer core groups that are defined for your system. You can also use this page to define a new peer core group or delete an existing peer core group.

To access this administrative console page, click **Servers > Core Groups > Core group bridge settings > Tunnel peer access points** *access_point__name* **> Peer core groups** . You can then click the name of a listed peer core group to view the peer bridge endpoints that have been associated with that peer core group.

You can also click **New** if you want to create a new peer core group, or you can select an existing peer core group and then click **Delete** to delete a peer core group.

Core group name

Specifies the name of the core group in the foreign cell where the end point resides.

Peer core group settings

Use this page to create a peer core group. Peer core groups are core groups that reside in different cell. The local core group bridge attempts to establish communication between peer core groups in the order in which they appear in the list of peer core groups.

To access this administrative console page, click **Servers** > **Core Groups** > **Core group bridge settings** > **Tunnel peer access points** *access_point__name* > **Peer core groups** > **New**.

Core group name

Specifies the name of the core group in the foreign cell where the end point resides.

Peer bridge endpoints

Specifies the host name and port of the peer core group. The peer bridge endpoint is used as a bridge interface to another peer core group. This bridge interface enables members of peer core groups to use the core group bridge service to communicate with other even though they reside in different cells.

One or more peer bridge endpoints can be specified for a peer core group.

Configuring the core group bridge between core groups that are in different cells

Use this task to configure communication between core groups that are in different cells.

Before you begin

Verify that the following conditions exist:

- You have two or more core groups that are in different cells. A core group is a statically defined component of the high availability manager.
- Any cell that uses core group bridges to connect to core groups in other cells have a name that is unique when compared to the names of the other cells.

About this task

A core group bridge should be used in situations where the availability status of servers in different cells needs to be shared across all of those cells. For example, you might have a situation where a WebSphere proxy server needs the ability to route requests to servers in other cells.

You can use core group bridge custom properties to set up advanced configurations for a core group bridge.

gotcha: When configuring core group bridges, remember the following requirements:

- Whenever a change is made in core group bridge configuration, including the addition of a new bridge, or the removal of an existing bridge, you must fully shutdown, and then restart all core group bridges in the affected access point groups.
- There must be at least one running core group bridge in each core group. If you configure two bridges in each core group, a single server failure does not disrupt the bridge functionality. Also, configuring two bridges enables you to periodically cycle out one of the bridges. If all the core group bridges in a core group are shutdown, the core group state from all foreign core groups is lost.

bprac: It is also recommended that:

- Core group bridges be configured in their own dedicated server process, and that these processes have their monitoring policy set for automatic restart.
- For each of your core groups, you set the IBM_CS_WIRE_FORMAT_VERSION core group custom property to the highest value that is supported on your environment.
- To conserve resources, do not create more than two core group bridge interfaces when you define a core group access point. You can use one interface for workload purposes and another interface for high availability. Ensure that these interfaces are on different nodes for high availability purposes. For more information, see the frequently asked question information on core group bridges.
- You should typically specify ONLY two bridge interfaces per core group. Having at least two bridge interfaces is necessary for high availability. Having more than two bridge interfaces adds unnecessary overhead in memory and CPU.

To configure a core group bridge between core groups in different cells, complete the following procedure for each of the cells in your configuration.

Procedure

1. Configure bridge interfaces for your core group access point. Configuring a bridge interface indicates that the specified node, server, and chain combination is a core group bridge server. This node and server use the specified chain to communicate with other core groups.
   a. In the administrative console, click **Servers > Core Groups > Core group bridge settings > Access point groups >***access_point_group* **> Core group access points**
   b. Select one of the listed core group access points. Then click **Show detail > Bridge interfaces > New**.
   c. Select a node, server, and transport chain for your bridge interface.
   d. Click **Apply**.
   e. Repeat this set of steps to add more bridge interfaces to the core group access point.

      Define at least two bridge interfaces for each core group access point to back up your configuration. When you define two core group bridge servers, if one of these two servers fails, the other server handles any pending communication, thereby preventing an interruption in the communication between the core groups.

      gotcha: The bridge interfaces that you select must all have the same transport chain.
2. If you want the ability to add core group bridge servers to the configuration without restarting the other servers in the configuration, define the cgb.allowUndefinedBridges custom property on all of the access point groups in your configuration.
   a. In the administrative console, click **Servers > Core Groups > Core group bridge settings > Access point groups >***access_point_group* **> Custom properties > New**.
   b. Type the name as cgb.allowUndefinedBridges and set the value to any string.

The existence of the cgb.allowUndefinedBridges property automatically enables this property. Therefore, you can set the value to any string value. Setting the value to `false` does not disable the property. To disable the property, you must remove it from the list of defined custom properties or change its name.

c. Click **Apply** and save your configuration.

When you complete this step on all the access point groups in your configuration, you can add a bridge interface to one of the cells. You can save the configuration so that it is propagated to all of the nodes. Instead of restarting all of the application servers, you need to restart the new bridge interface server only.

3. Add peer access points and peer ports to your access point group.

   If you defined the cgb.allowUndefinedBridges custom property for all of the access point groups in your configuration, you do not need to add peer access points or peer ports to the listener cell.

   Add a peer access point for each core group that is in another cell. Within each peer access point, you should configure a peer port that corresponds to each bridge interface in the other cell. Before you add a peer access point, determine the following information about the other cell:

   - Cell name
   - Core group name
   - Core group access point name
   - Host and port information. The host and port correspond to the bridge interfaces that are configured in the other cell. Specify a peer port for each bridge interface that is in the other cell.

   a. In the administrative console, click **Servers > Core Groups > Core group bridge settings > Access point groups >***access_point_group* **> Peer access points > New**.

   b. Specify the information for your peer access point.

      In addition to specifying a name for the peer access point, you must complete the following actions:

      - Specify the remote cell where the peer access point resides.
      - Specify the name of the core group in the remote cell to which the peer access points belongs.
      - Select either **Use peer ports** or **Use proxy peer access points**, depending on whether the peer access point can be reached directly or can only be reached indirectly through another peer access point.
      - Select the level of access that you want a server from another cell to have to the local cell when that server uses this access point to establish communication with the local cell.
        - If you select **Full access**, the communicating server can read data from and write data to the local cell. This level of access is appropriate if there is no reason to restrict read or write access to the local cell.
        - If you select **Read only**, the communicating server can read data from the local cell, but is not permitted to write data to the local cell. This level of access is appropriate if applications running in other core groups need to access data that is contained in the local cell. but you want to prevent the communicating server from changing the data.
        - If you select **Write only**, the communicating server can write data to the local cell, but is not permitted to read data from the local cell. This level of access is appropriate if applications running in other core groups need to write data to the local cell, but the data stored on the local cell is sensitive. For example, the local cell might contain customer account numbers, and you do not want applications that resides outside of the local cell to read this information.

   c. Click **Next**.

   d. Select **Use peer ports**. Specify the host and port information for your peer cell. For example, if you defined a bridge interface in *cell_x*, use that configuration information for your peer port in *cell_y*.

   e. Click **Next** and then **Finish**. Save your configuration.

4. Optional: If more than one bridge interface is defined in your peer cell, add additional peer ports for each bridge interface.

   a. Click **Peer access points >** *peer_access_point* **> Show detail > Peer ports > New**.

   b. Enter the host name and port.
   c. Click **Apply** , and save your changes.

5. Optional: Configure the high availability manager protocol to establish transparent bridge failover support.

   During core group bridge state rebuilds, cross-core group state can be moved between running bridges. This situation might cause the data to be temporarily unavailable until the bridge has completed the rebuild process.

   If you are running on Version 7.0.0.1 or later, set the IBM_CS_HAM_PROTOCOL_VERSION core group custom property to `6.0.2.31` for all of your core groups to avoid a possible high availability state outage during core group bridge failover. When this custom property is set to `6.0.2.31`, the remaining bridges recover the high availability state of the failed bridge without the data being unavailable in the local core group.

   Complete the following actions to set the IBM_CS_HAM_PROTOCOL_VERSION core group custom property to `6.0.2.31` for all of your core groups.

   a. Shut down all core group bridges in all of your core groups.
   b. Repeat the following actions for each core group in each of your cells:
      1) In the administrative console, click **Servers > Core Groups > Core group settings >** *core_group_name* **> Custom properties**.
      2) Specify `IBM_CS_HAM_PROTOCOL_VERSION` in the **Name** field, and `6.0.2.31` in the **Value** field.
      3) Save your changes.
   c. Synchronize your changes across the topology.
   d. Restart all of the bridges in the topology.

   All of the core groups within this topology are using the 6.0.2.31 high availability manager protocol.

Results

You configured the core group bridge between core groups that are in different cells.

The following figure illustrates the resulting core group bridge between the two core groups that are in two different cells. Each cell has a defined access point group that contains one core group access point for the core group that is in the cell and a peer access point for the other cell.

Example

The following example illustrates the configuration steps that are performed to set up a core group bridge between two cells. In this example:

- The two cells are referred to as the primary cell and the remote cell.
- wasdmgr02/dmgr/DCS is the name of the deployment manager on the primary cell, and wasdmgr02/dmgr/DCS is the name of the deployment manager on the remote cell.
- wasna01/nodeagent/DCS is the name of a node on both the primary cell and the remote cell.
- CGAP_1/DefaultCoreGroup is the name of the core groups on both the primary cell and the remote cell.

1. Using the administrative console for the primary cell, click **Servers > Core groups > Core group bridge settings > Access point groups > DefaultAccessPointGroup > Core group access points.**
2. Select **CGAP_1/DefaultCoreGroup**. and then click **Show Detail**.
3. Select **Bridge interfaces**, and then click **New**.
4. In the **Bridge interfaces** field, select the deployment manager, **wasdmgr02/dmgr/DCS**, from the list of available bride interfaces, and then click **OK**.

5. Click **New** to create a second bridge interface.
6. In the **Bridge interfaces** field, select a node agent, such as **wasna01/nodeagent/DCS**, and then click **OK** to save your changes.
7. Go to the administrative console for the remote cell, and click **Servers > Core groups > Core group bridge settings > Access point groups > DefaultAccessPointGroup > Core group access points.**.
8. Select **CGAP_1/DefaultCoreGroup**. and then click **Show Detail**.
9. Select **Bridge interfaces**, and then click **New**.
10. In the **Bridge interfaces** field, select the deployment manager, **wasdmgr03/dmgr/DCS**, from the list of available bride interfaces, and then click **OK**.
11. Click **New** to create a second bridge interface.
12. In the **Bridge interfaces** field, select the node agent, **wasna01/nodeagent/DCS**, from the list of available bride interface, and then click **OK** to save your changes.
13. Save your changes.
14. Gather the following information for the remote cell:
    - The DCS port for the deployment manager. Click **System administration > Deployment manager > Ports > DCS_UNICAST_ADDRESS**, and write down the port number for DCS_UNICAST_ADDRESS. In this example, the DCS port for the deployment manager is 9353.
    - The DCS port for the wasna01 node agent. Click **System administration >Node agents > wasna01 > Ports > DCS_UNICAST_ADDRESS**, and write down the port number for DCS_UNICAST_ADDRESS. In this example, the DCS port for the node agent is 9454.
    - The name the of the core group in the cell to which the Enterprise Javabean (EJB) cluster belongs. Click **Servers > Core groups > Core group settings > DefaultCoreGroup >. Core group members**, verify that your servers are members of the DefaultCoreGroup core group, and then write down the core group name. In this example the core group name is DefaultCoreGroup.
    - The name of the cell. Click **System administration > Cell**, and then write down the name that displays in the **Name** field. In this example, the name of the name of the cell is wascell03.
    - The name of the core group access point. Click **Servers > Core groups > DefaultCoreGroup > Core group bridge settings**, expand the **DefaultAccessPointGroup** field, and write down the name of the core group access point that displays when you expand **Core Group DefaultCoreGroup**. In this example the name of the core group access point is CGAP_1.
15. Go back to the administrative console for the primary cell and gather the same information about the primary cell. In this example:
    - The DCS port for the deployment manager on the primary cell is 9352.
    - The DCS port for the wasna01 node agent on the primary cell is 9353.
    - The name of the core group in the cell to which the EJB cluster belongs is DefaultCoreGroup.
    - the name of the cell is wascell02.
    - The name of the core group access point is CGAP_1.
16. Create a new peer access point that points to the remote cell. In the primary cell administrative console, click **Servers > Core groups > Core group bridge settings > Access point groups > DefaultAccessPointGroup > Peer access points**.
    a. Click **New** to start the Create new peer access point wizard.
    b. Specify the name of the new peer access point, `RemoteCellGroup`, in the **Name** field, `wascell03` in the **Remote cell name** field, `DefaultCoreGroup` in the **Remote cell core group name** field, and `CGAP_1` in the **Remote cell core group access point name** field.
    c. Click **Next**, and then select either **Use peer ports** or **Use a proxy peer access point**. For this example, we select **Use peer ports** , and specify `washost02` in the **Host** field, and `9353` in the **Port** field. These values are the host name and DCS port number for the deployment manager on the remote cell.

d. Click **Next**, confirm that the information that you specified for the new peer access point is correct, and then click **Finish**.

17. Create a second peer access point for the node agent.
    a. Select the peer access point that you just created, **RemoteCellGroup/wascell03/ DefaultCoreGroup/CGAP_1**, and then click **Show Detail**.
    b. In the Peer addressability section, select **Peer ports**, and then click **Peer ports > New**.
    c. Specify `washost04` in the **Host** field, and `9454` in the **Port** field. These values are the host name and DCS port number for the node agent on the remote cell.
18. Click **OK** and then click **Save** to save your changes to the master configuration.
19. Go to remote cell administrative console, and click **Servers > Core groups > Core group bridge settings > Access point groups > DefaultAccessPointGroup > Peer access points > New** to start the Create new peer access point wizard and create peer access points in the remote cell.
    a. Specify the name of the new peer access point, `PrimaryCellGroup`, in the **Name** field, `wascell02` in the **Remote cell name** field, `DefaultCoreGroup` in the **Remote cell core group name** field, and `CGAP_1` in the **Remote cell core group access point name** field.
    b. Click **Next**, and then select either **Use peer ports** or **Use a proxy peer access point**. For this example, we select **Use peer ports** , and specify `washost01` in the **Host** field, and `9352` in the **Port** field. These values are the host name and DCS port number for the deployment manager on the primary cell.
    c. Click **Next**, confirm that the information that you specified for the new peer access point is correct, and then click **Finish**.
20. Create a second peer access point for the node agent on the primary cell.
    a. Select the peer access point that you just created, **PrimaryCellGroup/wascell02/ DefaultCoreGroup/CGAP_1**, and then click **Show Detail**.
    b. In the Peer addressability section, select **Peer ports**, and then click **Peer ports > New**.
    c. Specify `washost03` in the **Host** field, and `9353` in the **Port** field. These values are the host name and DCS port number for the node agent on the primary cell.
21. Click **OK** and then click **Save** to save your changes to the master configuration.
22. Restart both cells.

What to do next

Continue configuring the high availability environment.

Core group bridge settings

The core group bridge is the service that enables communication between core groups. A core group is a statically defined component of the high availability manager. Use this page to view the structure of your access point groups and tunnel access point groups.

To view this administrative console page, click **Servers > Core Groups > Core group bridge settings**.

An access point group links core groups that are in the same cell or in different cells. The access points that are defined for an access point group enable these core groups to communicate with each other. If one or more of the core groups reside outside of a firewall, you must also define a tunnel access point group, and at least one tunnel access point for each core group that resides outside of the firewall.

Each access point group is a collection of core group access points. Each tunnel access point group is a collection of core group access points, and tunnel peer access points.

- If you are configuring communication between core groups that are in the same cell, configure an access point group with a core group access point for each core group in the cell.

- If you are configuring communication between core groups in different cells, configure an access point group that has one core group access point for the local cell and a peer access point for each other cell.
- If you are configuring communication between core groups, and one or more of the core groups reside outside of a firewall, configure a tunnel access point group with a core group access point for each core group that resides inside of the firewall, and a tunnel peer access point for each core group that resides outside of the firewall.

Each core group access point has one or more bridge interfaces. Each peer access point and tunnel peer access point has a proxy peer or one or more peer ports. A bridge interface is a server that is configured to communicate with other core groups by using a particular transport chain.

bprac: Even though it is possible to use a deployment manager as a core group bridge interface, it is recommended that you use an application server that is not a deployment manager.

- Click **Access point groups** to configure the settings for each access point group that is configured.
- Click **Tunnel access point groups** to configure the settings for each tunnel access point group that is configured.
- Click **Tunnel peer access points** to configure the tunnel peer access points for a tunnel access point group.
- Click **Tunnel templates** to use one of the tunnel templates that are provided with the product, to define a tunnel access point group and the tunnel access points for that group.
- **Access point group** - An access point group defines the core groups that need to communicate with each other. Each access point group consists of a collection of core groups.
  - **Core group** - Specifies a core group that is in this access point group. Core groups are referenced by core group access points.
    - **Core group access point** - The core group access point defines the set of servers that provide access to the core group. Bridge interfaces define the servers that are in the core group access point.
      - **Bridge interface** - A bridge interface defines a node, server, and chain for the core group access point. The chain defines the transport channels that the server uses for receiving information. Typically, all the bridge interfaces in a core group access point use the same chain.

        gotcha: The bridge interfaces listed in the display tree on this page are listed as `*,port`. The asterisk symbol represents the multi-home support for DCS instead of a specific hostname.
  - **Peer core group** - Specifies a core group in a different cell. Define peer access points to communicate with peer core groups.
    - **Peer access point** - Each peer access point is used to establish communication with another core group. Each peer access point corresponds to a core group access point that is in the peer cell. Use one or more peer ports or one proxy peer to define the communication settings.
      - **Peer port** - Each peer port identifies a bridge interface of a core group bridge in the peer cell.
      - **Proxy peer** - A proxy peer is used to identify the communication settings for a peer access point that cannot be accessed directly through peer ports. A proxy peer specifies a peer access point that can communicate with the destination core group. The specified proxy peer must be a peer access point that has defined ports.
- **Tunnel access point group** - A tunnel access point group defines the core groups that reside outside of the firewall.
  - **Core group** - Specifies a core group that is in this tunnel access point group. These core groups are referenced by core group tunnel access points.
    - **Core group access point** - The core group access point defines the set of servers that provide access to the core group. Bridge interfaces define the servers that are in the core group access point.

    - **Bridge interface** - A bridge interface defines a node, server, and chain for the core group access point. The chain defines the transport channels that the server uses for receiving information. Typically, all the bridge interfaces in a core group access point use the same chain.

      gotcha: The bridge interfaces listed in the display tree on this page are listed as `*,port`. The asterisk symbol represents the multi-home support for DCS instead of a specific hostname.

  - **Tunnel peer access point** - The tunnel peer access point defines the set of servers, that reside outside of the firewall, that provide access to the core group. Bridge interfaces define the servers that are in the tunnel peer access point.
    - **Bridge interface** - A bridge interface defines a node, server, and chain for the core group access point. The chain defines the transport channels that the server uses for receiving information. Typically, all the bridge interfaces in a core group access point use the same chain.

      gotcha: The bridge interfaces listed in the display tree on this page are listed as `*,port`. The asterisk symbol represents the multi-home support for DCS instead of a specific hostname.

Access point group collection

Use this page to view the sets of access point groups. Access point groups define the set of core groups that communicate with each other. Access point groups that connect multiple cells must have one core group access point and a single peer access point for each remote cell. Access point groups that provide communications between core groups in the same cell must contain only core group access points.

To view this administrative console page, click **Servers > Core Groups > Core group bridge settings > Access point groups**.

Name

Specifies the name of the access point group. The access point group name must be unique within the cell.

Access point group settings

Use this page to modify the core group access points and the peer access points that belong to an access point group. An access point group defines the set of core groups that can communicate with each other.

To view this administrative console page, click **Servers > Core Groups > Core group bridge settings > Access point groups >** *access_point_group_name*.

From this page, you can edit the core group access points or the peer access points that belong to the access point group.

Name:

Specifies the name of the access point group. The access point name must be unique within a cell.

Member communication key:

Specifies the name that core group members use to establish communication with other core group members that are defined as part of this tunnel access point group.

The default value is the name of the access point group.

Peer access point selection:

Use this page to control which peer access points are associated with this access point group. You can also use this page to create new peer access points for this access point group, or delete existing peer access points.

To view this administrative console page, click **Servers** > **Core Groups** > **Core group bridge settings** > **Access point groups** > *access_point_group_name* > **Peer access points**.

To see the configuration settings for a peer access point, select that peer access point, and then, click **Show details** for the list that contains that peer access point.

To create a new peer access point, click **New**.

To add an existing peer access point to this access point group, select one of the available peer access points, and click the first arrow to move that access peer point to the Peer access points list for this access point group.

To remove a peer access point from this access point group, select one of the peer access points from the list of peer access points that are contained in this access point group, and then click the second arrow to move the peer access point back to the Available peer access points list.

To delete a peer access point, select the peer access point that you want to delete, and click **Delete**.

When you finish adjusting your access points, click **OK** , and then click **Save** to save and synchronize your changes with all managed nodes.

Available peer access points:

Specifies a list of defined peer access points that are available for you to add to this access point group.

Peer access points in access_point_group: Specifies a list of peer access points that are in this access point group.

Core group access point collection

Use this page to configure your set of core group access points. Core group access points define the set of servers that provide access to the core group. At least one core group access point must be defined for each core group in the local cell.

To access this administrative console page, click **Servers** > **Core Groups** > **Core group bridge settings** > **Access point groups** > *access_point_group_name* > **Core group access points**.

Available core group access points

A list of core group access points that are available to add to the access point group.

Core group access points in *access point group*

A list of core group access points that are in the specified *access point group*.

Core group access point settings

Use this page to configure your core group access points. The core group access point defines the set of core group bridge servers that provide access to the core group. A core group bridge server is an application server that is configured to run the core group bridge service. Each core group access point has a collection of bridge interfaces. See the topic *Bridge interface settings* for more information about bridge interfaces.

Typically, you should not have to manually create a core group access point because a core group access point is automatically created whenever you create a core group. If multiple core group access points exist for a core group, then each core group access point must have the same set of bridge interfaces. For example, given the following situation:

- Core group access points CGAP_DF1 and CGAP_DF2 both exist for the DefaultCoreGroup core group.
- Access point CGAP_DF1 has server1 as a bridge interface.
- Access point CGAP_DF2 has server1 and server2 as bridge interfaces,

Server1 and server2 do not function correctly, and an error message similar to the following message displays in `SystemOut.log`:

```
CWRCB0201E: The core group bridge service could not be started due to a configuration error:
   Core group access points [DefaultCoreGroup:CGAP_DF1] and [DefaultCoreGroup:CGAP_DF2]
   must have bridge interfaces on the same set of node/server combinations.
```

Note: This topic references one or more of the application server log files. Beginning in WebSphere Application Server Version 8.0 you can configure the server to use the High Performance Extensible Logging (HPEL) log and trace infrastructure instead of using `SystemOut.log` , `SystemErr.log`, `trace.log`, and `activity.log` files or native z/OS logging facilities. If you are using HPEL, you can access all of your log and trace information using the LogViewer command-line tool from your server profile bin directory. See the information about using HPEL to troubleshoot applications for more information on using HPEL.

To access this administrative console page, click **Servers** > **Core Groups** > **Core group bridge settings** > **Access point groups** > ***access_point_group_name*** > **Core group access points** > ***core_group_access_point_name*** > **Show details**.

Name:

Specifies the name for the core group access point.

Core group:

Specifies the core group that is associated with this core group access point.

Bridge interface collection

Each core group access point has a collection of bridge interfaces. This collection defines the interfaces on the set of servers that provide access to the core group. All the servers in this collection have the core group bridge service enabled.

The core group bridge service provides communication between core groups. A bridge interface defines the node, the server, and the chain for the core group access point. The chain defines the transport channels that are used by the server for receiving information.

To access this administrative console page, click **Servers > Core Groups > Core group bridge settings > Access point groups >** *access_point_group_name* **> Core group access points >** *access_point_name* **> Bridge interfaces**. Click **New** on this page if you want to create a new bridge interface.

Server

Specifies the node and the server combinations that are bridge interfaces for the core group access point.

Node

Specifies the node on which the cluster member is running.

Transport channel chain

Specifies the name of the transport chain that is used for transport by the bridge interface. For all the core group access points in an access point group, the transport chains must resolve to the same host. To ensure that the transport chains resolve to the same host, use the same chain for all of the core group access points in an access point group.

Bridge interface settings

Use this page to specify the bridge interfaces that provide access to the core group access point. A bridge interface is a server that runs the core group bridge service. The core group bridge service is the service that establishes communication between core groups.

bprac:

- A server that you select as a core group bridge interface should not be a server that handles production responsibilities, such as filtering requests to cluster members, or WebSphere proxy servers. Any server that is a core group bridge interface experiences extensive memory and CPU usage during core group bridge startup, and failover processing, which occurs when one of the bridges in a core group stops. If you cannot dedicate a server as a core group bridge interface, you should select a node agent as your core group bridge interface.
- You should typically specify ONLY two bridge interfaces per core group. Having at least two bridge interfaces is necessary for high availability. Having more than two bridge interfaces adds unnecessary overhead in memory and CPU.

To access this administrative console page, click **Servers > Core Groups > Core group bridge settings > Access point groups >** *access_point_group_name* **> Core group access points >** *access_point_access_point_name* **> Show detail > Bridge interfaces >** *server_node*.

Bridge interfaces: Select the servers, that are available in the appropriate core group, that you want to use as the bridge interfaces for your core group access point.

Bridge interface creation

A bridge interface specifies a particular node and server that runs the core group bridge service. A bridge interface is defined by a unique combination of a node, a server, and a transport chain. A transport chain represents a network protocol stack that is operating within an application server.

To access this administrative console page, click **Servers > Core Groups > Core group bridge settings > Access point groups >** *access_point_group_name***> Core group access points >** *access_point_name* **> Show detail > Bridge interfaces > New**.

Available bridge interfaces:

Specifies the node, server and transport chain combinations that are available to become bridge interfaces for this core group access point. Only bridge interfaces with transport chains that contain Transmission Control Protocol (TCP) inbound channels using the same port name as existing bridge interfaces in this core group access point are displayed. The bridge interfaces that are already used by any core group access point are not displayed.

Peer access point collection

Use this page to view all of the peer access points that are defined on your system. You can include one or more of these access points in an access point group. Only one peer access point should be defined for a remote cell.

You can also use this page to create new peer access points, or delete an existing peer access point.

To access this administrative console page, click **Servers > Core Groups > Core group bridge settings > Peer access points**.

Name

Specifies the name of the peer access point. The name must be unique within the local cell.

Remote cell name

Specifies the name of the remote cell in which the peer access point resides.

gotcha: This property is case sensitive. The value you specify must exactly match the name of the cell in which the peer access point resides. For example, if WASCell05 is the name of the cell that contains the peer access point, you must specify `WASCell05` as the value for this property. If you specify `wascell05` as the value for this property, communication between the two core groups fails.

Remote cell core group name

Specifies the name of the core group in the remote cell to which the peer access point belongs.

Remote cell core group access point name

Specifies the name of the core group access point that is in the remote cell.

Peer access point settings

Use this page to configure a peer access point. Each peer access point is used to communicate with core groups in other cells. A peer access point corresponds to a core group access point in the peer cell. The peer access point communication settings are specified by using one or more peer endpoints or a proxy peer.

A peer access point must contain either peer ports or a proxy peer access point, but not both. When the peer access point is directly accessible within its access point group, specify peer ports. When the peer access point can be reached only indirectly, use a proxy peer access point. A proxy peer access point is used to identify the communication settings for the peer access point that cannot be accessed directly. The proxy peer access point specifies a peer access point that can communicate with the appropriate destination core group. The specified proxy peer access point must be a peer access point that has defined ports.

To view this administrative console page, click **Servers** > **Core Groups** > **Core group bridge settings** > **Access point groups** > *access_point_group_name* > **Peer access points** > *peer_access_point_name* > **Show detail**.

Name:

Specifies the name of the peer access point. The name must be unique within the local cell.

Remote cell name:

Specifies the name of the remote cell in which the peer access point resides.

gotcha: This property is case sensitive. The value you specify must exactly match the name of the cell in which the peer access point resides. For example, if WASCell05 is the name of the cell that contains the peer access point, you must specify `WASCell05` as the value for this property. If you specify `wascell05` as the value for this property, communication between the two core groups is not established.

Remote cell core group name:

Specifies the name of the core group in the remote cell to which the peer access point belongs.

Remote cell core group access point name:

Specifies the name of the core group access point that is in the remote cell.

default	defaultCoreGroupAccessPoint

Use peer ports:

When selected, specifies that you are using peer ports instead of a proxy peer access point. Use peer ports when the peer access point is directly accessible within its access point group. Click **Peer ports** to specify the peer ports for the peer access point.

Use proxy peer access point:

When selected, specifies that you are using a proxy peer access point instead of peer ports. A proxy peer is defined when the peer access point can be reached only indirectly through another peer access point. A proxy peer is used to identify the communication settings for the peer access point that cannot be accessed directly. The proxy peer specifies a peer access point that can communicate with the destination core group. The specified proxy peer must be a peer access point that has defined peer ports.

When you select this type of peer addressability, you must also select a proxy peer access point from the list of available proxy peer access points.

Cell-level access:

Specifies the level of access that a server from another cell is given to the local cell when that server uses this access point to establish communication with the local cell.

- **Full access** enables the communicating server to read data from and write data to the local cell. This level of access is appropriate if there is no reason to restrict read or write access to the local cell.
- **Read only** enables the communicating server to read data from the local cell, but prevents that server from writing data to the local cell. This level of access is appropriate if applications running in other core groups need to access data that is contained in the local cell but you want to make sure that the data stored on the local cell is not changed.
- **Write only** enables the communicating server to write data to the local cell, but prevents that server from reading data from the local cell. This level of access is appropriate if applications running in other core groups need to write data to the local cell, but the data stored on the local cell is sensitive. For example, the local cell might contain customer account numbers, and you do not want applications that resides outside of the local cell to read this information.

Peer port collection

Use this page to define the peer ports for the peer access point. Each peer port identifies a bridge interface of a core group bridge service in the peer cell. Each peer access point that does not have a proxy peer must have one or more peer ports.

To view this administrative console page, click **Servers > Core groups > Core group bridge settings > Access point groups >** *access_point_group_name* **> Peer access points >** *peer_access_point_name* **> Show detail > Peer ports**.

Host

Specifies the host name that is used by the bridge interface in the remote cell.

Port

Specifies the port that is used by the bridge interface in the remote cell.

Peer port settings

Use this page to configure a peer port. A peer port identifies the host name and port of an application server that is a bridge interface in another cell. This application server is using the core group bridge service to communicate with other core groups. Each peer access point can have one or more peer ports. Each port identifies a bridge interface of a core group bridge service in the peer cell.

To view this administrative console page, click **Servers** > **Core Groups** > **Core group bridge settings** > **Access point groups** > *access_point_group_name* > **Peer access points** > *peer_access_point_name* > **Show detail** > **Peer ports** > *peer_port_name* .

Host:

Specifies the host name on which the core group bridge in the remote cell is listening. If the remote bridge is on a multi-homed system, enter a dotted decimal IP address.

Port:

Specifies the port number that is associated with the host on which the core group bridge in the remote cell is listening.

Core group bridge custom properties

Use these custom properties for advanced configurations for core groups and core groups that communicate with the core group bridge.

You can use the custom properties page to define the following core group bridge custom properties:

- "CGB_ENABLE_602_FEATURES"
- "cgb.allowUndefinedBridges"
- "cgb.rebuild.waitTime" on page 128
- "FW_PASSIVE_MEMBER" on page 128
- "IBM_CS_LS_DATASTACK_MEG" on page 129

CGB_ENABLE_602_FEATURES

depfeat: You should use the cgb.allowUndefinedBridges custom property instead of the CGB_ENABLE_602_FEATURES custom proeprty. The cgb.allowUndefinedBridges custom property performs the same functionality as the CGB_ENABLE_602_FEATURES custom property performed in Versions 6.0.x and 6.1.x.

If you have automation that adds this property to your core group bridge settings, this automation still works in this version of the product. However you should update your automation to add the cgb.allowUndefinedBridges custom property instead of the CGB_ENABLE_602_FEATURES custom property because the CGB_ENABLE_602_FEATURES custom property might not be recognized as a valid property in future releases.

cgb.allowUndefinedBridges

You can define the cgb.allowUndefinedBridges custom property on all of the access point groups in your configuration if you want to have the capability to add core group bridge servers to the configuration without restarting the other servers in the configuration. After you enable this property, you can add a core group bridge server in one cell, without modifying the configuration in the other cell to include peer ports for the core group bridge server. The core group in the other cell automatically discovers the peer ports for the first cell when the first cell initiates communication with the second cell.

transition: The cgb.allowUndefinedBridges custom property performs the same functionality as the CGB_ENABLE_602_FEATURES custom property performed in Versions 6.0.x and 6.1.x. If you have automation that adds this property to your core group bridge settings, this automation still works in this version of the product. However you should update your automation to add the cgb.allowUndefinedBridges custom property instead of the CGB_ENABLE_602_FEATURES custom property because the CGB_ENABLE_602_FEATURES custom property might not be recognized as a valid property in future releases.

The existence of the cgb.allowUndefinedBridges property automatically enables this property. Therefore, you can set the value to any string value. Setting the value to `false` does not disable the property. To disable the property, you must remove it from the list of defined custom properties or change its name.

If you decide to use the cgb.allowUndefinedBridges custom property, you must decide which cell is the listener cell and which cell initiates communication with the other cells before you begin your configuration. The listener cell does not need to contain any peer access points or peer ports for the other cells in the configuration. For example, in a configuration that contains a secured cell and an unsecured cell, configure the unsecured cell as the listener. The unsecured cell cannot access information about the secured cell. Configure the core group bridge service on the listener cell first.

gotcha: Do not configure the cgb.allowUndefinedBridges custom property if you have already configured the FW_PASSIVE_MEMBER custom property. Any server for which the FW_PASSIVE_MEMBER custom property is configured cannot initiate contact with other systems in the configuration.

cgb.rebuild.waitTime

Use this property in a core group bridge configuration to specify, in seconds, the length of time that the core group bridge service waits for a core group bridge rebuild to complete.

This property is ignored unless you have specified the IBM_CS_HAM_PROTOCOL_VERSION=6.0.2.31 core group custom property for every core group that is part of the core group bridge configuration.

FW_PASSIVE_MEMBER

Use this property in a core group bridge configuration when there is a firewall between the core groups and the secure side of the firewall is configured to listen only.

Set the FW_PASSIVE_MEMBER custom property to make the bridge interfaces that are in the core group access point passive. Set the value on the core group access point that is on the secure side of the firewall so that the core group bridge interfaces listen for connections from the unsecured side of the firewall but do not initiate any connections. The servers on the secure side of the firewall are passive. The custom property should correspond to your defined firewall rules that allow connections from the unsecured region to the secure region only.

To configure this custom property, click **Servers** > **Core Groups** > **Core group bridge settings** > **Access point groups** > *access_point_group_name* > **Core group access points** > *core_group_access_point_name* > **Show detail** > **Custom properties** > **New** in the administrative console.

You also must set this custom property in any peer access points that refer to the core group access points that you configure with this custom property.

Figure 10. Configuring the FW_PASSIVE_MEMBER custom property

For example, *server_A* and *server_B* are configured in *core_group_1*. *Server_C* and *server_D* are configured in *core_group_2*. *Core_group_2* is behind a firewall that is configured to listen only through the firewall. *Core_group_1* is on the unsecured side of the firewall. *Core_group_1* and *core_group_2* can communicate with each other through an access point group. To configure *server_C* and *server_D* to be passive, perform the following steps:

1. In the administrative console for the cell that contains *core_group_2*, click **Servers** > **Core Groups** > **Core group bridge settings** > **Access point groups** > *access_point_group_name* > **Core group access points** > *core_group_access_point_name* > **Show detail** > **Custom properties** > **New**.
2. Add the FW_PASSIVE_MEMBER custom property. Enter any value to enable the property.
3. In the administrative console for the cell that contains *core_group_1*, click **Servers** > **Core Groups** > **Core group bridge settings** > **Access point groups** > *access_point_group_name*> **Peer access points** > *peer_access_point_name* > **Show detail** > **Custom properties** > **New**. The peer access point you select should correspond to the core group access point for *core_group_2*.
4. Add the FW_PASSIVE_MEMBER custom property. Enter any value to enable the property.

By configuring the FW_PASSIVE_MEMBER custom property, you configured the servers on the secured side of the firewall, *server_C* and *server_D*, to be passive. These servers listen for connections from the other side of the firewall but do not initiate any connections to the unsecured side of the firewall.

IBM_CS_LS_DATASTACK_MEG

Use this custom property to eliminate a condition that is reported by a message that is displayed repeatedly in your SystemOut.log file.

Note: This topic references one or more of the application server log files. Beginning in WebSphere Application Server Version 8.0 you can configure the server to use the High Performance Extensible Logging (HPEL) log and trace infrastructure instead of using SystemOut.log , SystemErr.log, trace.log, and activity.log files or native z/OS logging facilities. If you are using HPEL, you can access all of your log and trace information using the LogViewer command-line tool from your server profile bin directory. See the information about using HPEL to troubleshoot applications for more information on using HPEL.

You might see a message similar to the following message in the SystemOut.log file multiple times:

```
[9/24/04 13:28:19:497 CDT] 00000013 VSync         W
DCSV2005W: DCS Stack DefaultAccessPointGroup.P at Member 172.16.1.86:9353:
The amount of memory available for synchronization is low. The configured memory
size is 418816 bytes. Currently used memory size is 420307 bytes.
```

If the member IP address is in the format of a dotted decimal IP address and port, you can eliminate these messages by increasing the amount of memory that is allocated to the core stack that is used for core group communication. Increase the value of this property until you no longer see the message in your SystemOut.log file. Because the memory is dynamically allocated, setting a larger stack size than you need does not cause memory problems.

Set the custom property on the bridge interface that contains the particular member that is in the messages. You can also set the custom property on the access point group or the core group access point. If you set the value on the access point group or core group access point, all the bridge interfaces that are in the particular group are affected. If you set the value on an individual bridge interface and an access point group or core group access point, the value that is set for the bridge interface is used. If the value is set on both an access point group and a core group access point, the value that is set for the core group access point is used.

To configure this custom property, complete the following steps:

1. Set the custom property in the administrative console.
   - To set the custom property on a bridge interface, click **Servers** > **Core Groups** > **Core group bridge settings** > **Access point groups** > *access_point_group_name* > **Core group access points** > *core_group_access_point_name* > **Show detail** > **Bridge interfaces** > *bridge_interface_name*> **Custom properties** > **New** .

- To set the custom property on a core group access point, click **Servers** > **Core Groups** > **Core group bridge settings** > **Access point groups** > *access_point_group_name* > **Core group access points** > *core_group_access_point_name* > **Show detail** > **Custom properties** > **New**.
- To set the custom property on an access point group, click **Servers** > **Core Groups** > **Core group bridge settings** > **Access point groups** > *access_point_group_name* > **Show detail** > **Custom properties** > **New**.

2. Add the IBM_CS_LS_DATASTACK_MEG custom property. Enter a value that is greater than the default value of 5 megabytes.

Table 12. Units and default. The following table describes units and defaults for the setting.

Units	megabytes
Default	5

Chapter 14. Updating high availability applications in a z/OS environment

Application update procedure in a high availability environment

Application update involves distributing new application binaries to each of the servers in a cluster during configuration synchronization.

Figure 11. Application steady state - configuration view. This figure illustrates the configuration view of an application in steady state.

This distribution occurs via HTTP and happens even if the servers and the deployment manager are all located on the same LPAR. If the **synchronize with nodes** option is checked when the application update is saved, the synchronization request is sent to each node.

Normally, upon receiving the request, each node asynchronously orchestrates the synchronization process with the deployment manager. During this synchronization process, the application's binaries are downloaded from the deployment manager and stored on the node in the designated location (for example, `installedApps`).

The act of storing the new binaries triggers a configuration change event listener which then stops and restarts the application. Depending on such variables as dispatcher behavior, LPAR weighting, etc., a variance in the order and pace at which each server makes the new application available may be observed.

Because of the concurrent, asynchronous nature of node synchronization, continuous availability of the application being updated is not guaranteed. This is because there is no correspondence between the actual state of the application and the workload routing mechanisms. Client requests may be routed to a server even if that particular application is temporarily unavailable.

In a high availability environment, an application must remain available even during an update process. Therefore, application rollout to each cluster member, as well as workload routing to the cluster members must be carefully controlled to prevent workload from being routed to a cluster member that is undergoing the update process. When these two aspects are carefully controlled, an update can be installed on each cluster member without client requests arriving at a cluster member during the update process.

High availability configuration

The objective of any high availability configuration is to eliminate all single points of failure (SPOFs).

Figure 12. High Availability Configuration. This figure illustrates the recommended product configuration for high availability. The key elements are described in the text that accompanies this figure.

Following are the key elements of a high availability configuration:

- Network path redundancy leading up to the web servers and Applications Servers.

- Redundant web servers. (There must be at least two logical partitions (LPARs) in a high availability sysplex configuration.)
- A highly available sysplex configuration. These LPARs should be on separate hardware instances to eliminate hardware and software Single Points of Failures (SPOFs).
- A node on each LPAR that is configured into a WebSphere Application Server, Network Deployment cell. The deployment manager server (required, and configured on its own node) can be configured on either LPAR or on a separate LPAR. (The deployment manager server is not depicted in the preceding figure.) Also note, there is a daemon process (WebSphere CORBA Location Service) on each LPAR that has one or more nodes in the same cell.
- An application server defined on each node, and formed into a server cluster with the other application servers in the network.
- A dynamic virtual IP address (DVIPA) defined through the z/OS Sysplex Distributor as the daemon IP name for the cell. This IP address enables WLM-balanced routing and fail over between the LPARs for IIOP requests.
- A dynamic virtual IP address (DVIPA) defined through Sysplex Distributor as the HTTP transport name for the cell. This IP address enables WLM-balanced routing and fail over between the LPARs for sessionless HTTP requests.
- A static IP address is required for each node as an auxiliary HTTP transport name for the cell. This enables directed HTTP routing for sessional HTTP requests.
- A WebSphere web server plug-in must be installed in each of the web servers and configured to use the HTTP DVIPA for sessionless requests, and the static IP addresses for sessional requests.
- If using HTTP sessions, session state must be shared between cluster member using the data replication service (DRS) or session data must be stored in DB2®. If you are using stateful session Enterprise JavaBeans, the stateful session persistent store must be configured on a shared HFS. (Using stateful session Enterprise JavaBeans is not a best practice.)

Stopping an application server to manually update a high availability application

For manual application rollout, workload routing is controlled by stopping the application server on which the cluster member being updated resides. This results in a quiesce of that server. All existing requests already in the server are allowed to complete, but no new requests are accepted. Both the sysplex distributor and the WebSphere Application Server web server plug-in routes work away from the quiescing server. After all work has completed, you start the application update process on this server.

Before you begin

Determine which application servers are hosting the cluster members that need updating.

About this task

If you have a high availability application whose updates you want to manually control you can use this process or you can use the MVS™ Modify command to pause the listeners for the affected application servers. See the information about pausing an application server to manually update a high availability application.

To manually control application rollout and workload routing in a high availability environment:

Procedure

1. Disable all forms of automatic synchronization, across all nodes in the cell and save the changes. Perform one of the following processes to complete this step:
   - In the administrative console:

a. Click **System Administration > Node agents >** *node_agent_name* **> File Synchronization Service**.

b. Unselect the **Automatic Synchronization** and **Startup Synchronization** options.

c. Select the **Synchronize changes with nodes** option.

d. Click **Save**.

- Use wsadmin scripting to specify the following commands and then restart all affected node agents:

```
set node NODE
set na_id [$AdminConfig getid /Node:$node/Server:nodeagent/]
set syncServ [$AdminConfig list ConfigSynchronizationService $na_id]
$AdminConfig modify $syncServ {{autoSynchEnabled false}}
$AdminConfig modify $syncServ {{synchOnServerStartup false}}
$AdminConfig save

set nodeSync [$AdminControl completeObjectName type=NodeSync,node=$node,*]
$AdminControl invoke $nodeSync sync
```

gotcha: For a production environment, it is reasonable to always run the node agent with automatic synchronization disabled. However, it is advisable for startup synchronization to be enabled for the node agent so that it can acquire configuration updates that occur when the node agent is down. Startup Synchronization can be left enabled provided you can ensure that you will not restart the node agent manually, through automation, or through automatic restart manager during the application update process.

2. Update the application in the master configuration repository on the deployment manager server. Perform one of the following processes to complete this step:

- In the administrative console:

a. Click **Applications** > **Enterprise Applications**.

b. Select the application you want to update.

c. Complete the application update process.

d. Save your changes to the master configuration. **DO NOT** select the **Synchronize changes with nodes** option .

- Use wsadmin scripting to issue the following command:

```
set app_loc /path/to/app
set app_options {application options ie: -appname app}
set options [list -update]  lappend options $app_options
$AdminApp install $app_loc $options
$AdminConfig save
```

At this point, you have the updated the version of your application (App v2 in the following figure) in your Master Configuration. However, the original version of your application (App v1 in the following figure) is still running in the cluster that has Cluster members on LPAR1 and LPAR2.

Figure 13. Install application update. This figure illustrates the first stage of an application update in a high availability environment.

3. Stop the Application Server on LPAR1 and manually synchronize the node to the updated version of the application. This step may take time to complete because the server must wait for all currently assigned work items to complete before shutting down.

Perform one of the following processes to complete this step:

- In the administrative console:

a. Click **Servers > Server Types > WebSphere application servers**.

   b. Select the cluster member you want to stop and update. This cluster member should be on LPAR1.
   c. Click **STOP**. The Cluster Stop method should not be used, because it will stop all Servers within the cluster and the application will no longer be available.
- Use wsadmin scripting: to issue the following commands:

  ```
  set node NODE
  set server SERVER
  $AdminControl stopServer $server $node
  ```

- Issue the following command from the MVS Console:

  ```
   STOP short_server_name
  ```

  For example:

  ```
  STOP BBOS001
  ```

4. Synchronize the node. Perform one of the following processes to complete this step:
   - In the administrative console:
     a. Click **System Administration > Node agents**.
     b. Select the node you want to synchronize, and then click **Full Resynchronize**.
   - Use wsadmin scripting to issue the following commands:

     ```
     set node NODE
     set nodeSync [$AdminControl completeObjectName type=NodeSync,node=$node,*]
     $AdminControl invoke $nodeSync sync
     ```

   As illustrated in the following figure, the updated version of the application (App v2) now resides in the node on LPAR1.

Figure 14. Update the node on LPAR1. This figure illustrates the first stage of an application update in a high availability environment with two LPARs.

5. Restart the server on LPAR1. Perform one of the following processes to complete this step:
   - In the administrative console:
     a. Click **Servers > Server Types > WebSphere application servers**.
     b. Select the server you want to start, and then click **START**.
   - Use wsadmin scripting to issue the following commands:

     ```
     set node NODE
     set server SERVER
     $AdminControl startServer $server $node
     ```

   - Issue the following command from the MVS Console:

     ```
     START procname,JOBNAME=server_short_name.ENV=cell_short_name.node_short_name.server_short_name
     ```

     For example:

     ```
     START BBO6ACR,JOBNAME=BBOS001,ENV=PLEX1.SY1.BBOS001
     ```

   When this server comes back up, it will be running the new version of the application (App v2),

Figure 15. Restart the server on LPAR1. This figure illustrates the completion of the first stage of an application update in a high availability environment.

6. With the new version of the application running on LPAR1, repeat the preceding three steps on the other LPARs in the cluster to update them with the new version of the application. The following figure illustrates what your configuration will look like in a two LPAR cluster.

Figure 16. Update the node on LPAR2. This figure illustrates the second stage of an application update in a high availability environment.

Results

The application update process is complete when the new version of the application is running on all of the cluster members in the cluster. The following figure illustrates what a two LPAR cluster will look like after you restart the server on LPAR2.

Figure 17. Restart server on LPAR2. This figure illustrates what a two LPAR cluster will look like after you restart the server on LPAR2.

Automatically rolling out updates to a high availability application

You can set up your system to perform automatic application rollout for your high availability applications. The automatic application rollout update process stops or pauses each application server that is hosting a cluster member that needs updating.

Before you begin

Determine which application servers are hosting the cluster members that need updating.

About this task

If you have a high availability application that frequently requires updates, you might want to automatically control the rollout of these updates.

When setting up the rollout update process you must decide whether you want the application servers to stop or pause while the update is made to an application. If you want the servers to pause, you must configure the node agent to allow the rollout update process to pause and resume the servers. You do not have to make any configuration changes if you want the rollout update process to stop and start the servers. However if the rollout update process stops and starts the servers, the process takes much longer to complete.

When an application server pauses, all requests already in the queue for that server are allowed to complete, but no new requests are accepted. Both the sysplex distributor and the WebSphere Application Server web server plug-in route work away from the server that is paused. After all of the requests assigned to that server complete, the application update process starts on that server.

When the update process is complete, the listener for that server resumes and the sysplex distributor and the WebSphere Application Server web server plug-in assign new work to that server. This process is repeated for all of the other servers in the cluster until all of the affected cluster members are updated.

To prepare your system to automatically rollout updates to a high availability application:

Procedure

1. Determine whether you want the rollout update process to stop or pause the affected application servers.
   - If you want the rollout update process to stop a server before it performs an application update, go to step 5.
   - If you want the rollout process to pause a server before it performs an application update, go to step 2. Steps 2, 3, and 4 are configuration changes that enable the rollout update process to pause and resume servers during an application update. You only have to make these changes once.
2. Add the **com.ibm.websphere.zos.mvsservices.enable** and **com.ibm.websphere.zos.rollout.pauseresume** custom properties to the node agent settings in the master configuration repository on the deployment manager server. These properties must be added to the settings for all of the node agents on which you want to automatically start the MVSServices MBean.
   a. In the administrative console, click **System Administration > Node agents >** *node_agent_name* **> Administration Services > Custom Properties > New**
   b. Enter **com.ibm.websphere.zos.mvsservices.enable** in the Name field, and **true** in the Value field.
   c. Click **Ok** .
   d. Click **New** .
   e. Enter **com.ibm.websphere.zos.rollout.pauseresume** in the Name field, and **true** in the Value field.
   f. Click **Ok** .
   g. Repeat these steps for any other node agents on which you want to automatically start the MVSServices MBean.
3. Click Save to save your changes directly to the master configuration.

   After you add the com.ibm.websphere.zos.rollout.pauseresume custom property and set it to `true`, any future application rollouts on this node are accomplished by pausing a listener for the application server, rather than stopping that application server.

   If the custom property com.ibm.websphere.zos.rollout.pauseresume is set to true, but the MVSServices MBean is not running on the configured node, the application servers on that node do not pause, and are not updated during the application update process.

   Messages are displayed on the MVS Console when an application server is paused or resumed similar to the messages that are displayed when an application server is stopped or started.
4. Restart the node agent. When you restart the node agent, the MVSServices MBean automatically starts.
5. Update the application configuration repository in the master on the deployment manager server.

   See the *Administering applications and their environment* PDF for more information.

Results

You are ready to start the rollout update process for an application that you need to update.

What to do next

To start the rollout update process, in the administrative console, click **Applications > Enterprise Applications**, select the application to update, and click **Rollout Update**.

gotcha: The application you select must reside on at least one member of a cluster.

This function automatically stops or pauses the server, updates the application, and then starts or resumes the server. Nodes are processed one at a time, so only the server residing on the node being processed is affected, the servers on the other nodes continue to process work. Eventually all of the nodes and servers are updated.

The update process is complete when the updated version of the application is running on all of the LPARs in the cluster.

Pausing an application server listener to manually update a high availability application

Instead of stopping the application server, you can use the MVS console Modify command to pause the listeners for that application server, perform the application update, and then resume the listeners. If you use this technique, you do not have to stop and then start the server to perform the application update.

Before you begin

Determine which application servers are hosting the cluster members that need updating.

About this task

If you have a high availability application whose updates you want to manually control, but you do not want to stop the affected servers, you can use the MVS Modify command to pause the listener for each of these servers and then update the application.

To pause the listeners and manually control application rollout in a high availability environment:

Note:

Procedure

1. Disable all forms of automatic synchronization, across all nodes in the cell and save the changes. Perform one of the following processes to complete this step:
   - In the administrative console:
     a. Click **System Administration** > **Node agents** > *node_agent_name* > **File Synchronization Service**.
     b. Deselect the **Automatic Synchronization** and **Startup Synchronization** options.
     c. Select the **Synchronize changes with nodes** option.
     d. Click **Save**.
   - Use wsadmin scripting to specify the following commands and then restart all affected node agents:

     ```
     set node NODE
     set na_id [$AdminConfig getid /Node:$node/Server:nodeagent/]
     set syncServ [$AdminConfig list ConfigSynchronizationService $na_id]
     $AdminConfig modify $syncServ {{autoSynchEnabled false}}
     $AdminConfig modify $syncServ {{synchOnServerStartup false}}
     $AdminConfig save

     set nodeSync [$AdminControl completeObjectName type=NodeSync,node=$node,*]
     $AdminControl invoke $nodeSync sync
     ```

   Note: For a production environment, it is reasonable to always run the node agent with automatic synchronization disabled. However, it is advisable for startup synchronization to be enabled for the node agent so that it can acquire configuration updates that occur when the node agent is down. Startup Synchronization can be left enabled provided you can ensure that you will not restart the node agent manually, through automation, or through automatic restart manager during the application update process.

2. Update the application in the master configuration repository on the deployment manager server. Perform one of the following processes to complete this step:
   - In the administrative console:
     a. Click **Applications > Enterprise Applications**.
     b. Select the application you want to update.
     c. Complete the application update process.
     d. Save your changes to the master configuration. **DO NOT** select the **Synchronize changes with nodes** option .
   - Use wsadmin scripting to issue the following command:

     ```
     set app_loc /path/to/app
     set app_options {application options ie: -appname app}
     set options [list -update]  lappend options $app_options
     $AdminApp install $app_loc $options
     $AdminConfig save
     ```

     At this point, you have the updated the version of your application (App v2 in the following figure) in your Master Configuration. However, the original version of your application (App v1 in the following figure) is still running in the cluster that has Cluster members on LPAR1 and LPAR2.

Figure 18. Install application update. This figure illustrates the first stage of an application update in a high availability environment.

3. Pause the listener of the application server on LPAR1 and manually synchronize the node to the updated version of the application. After you pause the listener, wait for all work items currently assigned to the server to complete, and then issue the following command from the MVS Console:

   MODIFY *short_server_name*,PAUSELISTENERS

   For example, if the short name for the server you are pausing is BBOS001, issue the following command:

   ```
   MODIFY BBOS001,PAUSELISTENERS
   ```

4. Synchronize the node. Perform one of the following processes to complete this step:
   - In the administrative console:
     a. Click **System Administration > Node agents**.
     b. Select the node you want to synchronize, and then click **Full Resynchronize**.
   - Use wsadmin scripting to issue the following commands:

     ```
     set node NODE
     set nodeSync [$AdminControl completeObjectName type=NodeSync,node=$node,*]
     $AdminControl invoke $nodeSync sync
     ```

   As illustrated in the following figure, the updated version of the application (App v2) now resides in the node on LPAR1.

Figure 19. Update the node on LPAR1. This figure illustrates the first stage of an application update in a high availability environment with two LPARs.

5. Resume the listener of the application server on LPAR1. Issue the following command from the MVS Console:

   MODIFY *short_server_name*,RESUMELISTENERS

For example, if the short name for the server you are pausing is BBOS001, issue the following command:

```
MODIFY BBOS001,RESUMELISTENERS
```

6. With the new version of the application running on LPAR1, repeat the preceding three steps on the other LPARs in the cluster to update them with the new version of the application. The following figure illustrates what your configuration will look like in a two LPAR cluster.

Figure 20. Update the node on LPAR2. This figure illustrates the second stage of an application update in a high availability environment.

Results

The application update process is complete when the new version of the application is running on all of the LPARs in the cluster.

Chapter 15. Troubleshooting high availability environments

High availability environment troubleshooting tips

Review the following topics if you encounter a problem with your high availability environment.

Message HMGR0218I is not displayed after a Java virtual machine starts

In a properly set up high availability environment, a high availability manager can reassess the environment it is managing and accept new components as they are added to the environment. For example, when a Java virtual machine (JVM) is added to the infrastructure, a discovery process begins. During startup the JVM tries to contact the other members of the core group. When it finds another running JVM, it initiates a join process with that JVM that determines whether or not the JVM can join the core group. If the new JVM is accepted as a member of the core group, all of the JVMs, including the new one, log message HMGR0218I . This message is also displayed on the administrative console.

Message HMGR0218I indicates the number of application servers in the core group that are currently online. If this message is not displayed after a JVM starts, either a configuration problem or a communication problem has occurred. To fix this situation, verify that the application server is running on a current configuration, by either using the deployment manager to tell the node agent to synchronize, or use the **syncNode** command o manually perform the synchronization. If the JVM still cannot join the core group, a network configuration problem exists.

Message HMGR0123I appears in the system log file

Message HMGR0123I might appear in the system log file if the status of core group members changes at the same time as the active coordinator changes. For example, this message might be issued when a core group member restarts and becomes the active coordinator.

This information message usually does not indicate a serious problem. Even if the message appears in the system log file, the new active coordinator receives the updated group status. If you want to minimize the occurrences of this message, you should select a core group member that does not frequently restart as the preferred core group coordinator.

CPU starvation messages in the system log file

`CPU starvation detected` error messages are displayed in the system log file whenever there is not enough physical memory available to allow the high availability manager threads to have consistent runtimes. When the CPU is spending the majority of its time trying to load swapped-out processes while processing incoming work, thread starvation might occur. The high availability manager detects this condition, and logs these error messages informing you that threads are not getting the required runtime.

To achieve good performance and avoid receiving these error messages, it is recommended that you allocate at least 512 MB of RAM for each Java process running on a single machine.

High CPU usage in a large cell configuration when security is enabled

With certain configurations and states, the amount of time spent in discovery becomes substantial.

- If a large the number of processes are defined within a core group, a proportionally large number of connections must be established to support these processes.
- If a large number of inactive processes are defined within a core group, a proportionally large number of connections are attempted during each discovery interval.
- If administrative security is enabled, the DCS connections are secured, and the impact of opening a connection greatly increases .

Use the Discovery and failure detection page in the administrative console to increase the length of time that the Discovery Protocol waits to calculates the set of unconnected core group members, and attempts to open connections to those members. Increasing the length of time between consecutive discovery periods decrease the amount of CPU time that is spent in discovery. Read the topic, *Configuring the Discovery Protocol for a core group*, for more information.

Transient high availability heartbeat failures under heavy load

Under heavy a workload, transient heartbeat failure conditions might occur between replication partners in an high availability configuration, even though both of the replication partners appear to be running properly.

Appendix. Directory conventions

References in product information to *app_server_root*, *profile_root*, and other directories imply specific default directory locations. This topic describes the conventions in use for WebSphere Application Server.

Default product locations - z/OS

app_server_root

Refers to the top directory for a WebSphere Application Server node.

The node may be of any type—application server, deployment manager, or unmanaged for example. Each node has its own *app_server_root*. Corresponding product variables are *was.install.root* and *WAS_HOME*.

The default varies based on node type. Common defaults are `configuration_root/AppServer` and `configuration_root/DeploymentManager`.

configuration_root

Refers to the mount point for the configuration file system (formerly, the configuration HFS) in WebSphere Application Server for z/OS.

The *configuration_root* contains the various *app_server_root* directories and certain symbolic links associated with them. Each different node type under the *configuration_root* requires its own cataloged procedures under z/OS.

The default is `/wasv8config/cell_name/node_name`.

plug-ins_root

Refers to the installation root directory for Web Server Plug-ins.

profile_root

Refers to the home directory for a particular instantiated WebSphere Application Server profile.

Corresponding product variables are *server.root* and *user.install.root.*

In general, this is the same as `app_server_root/profiles/profile_name`. On z/OS, this will always be `app_server_root/profiles/default` because only the profile name "default" is used in WebSphere Application Server for z/OS.

smpe_root

Refers to the root directory for product code installed with SMP/E or IBM Installation Manager.

The corresponding product variable is *smpe.install.root*.

The default is `/usr/lpp/zWebSphere/V8R0`.

Notices

References in this publication to IBM products, programs, or services do not imply that IBM intends to make these available in all countries in which IBM operates. Any reference to an IBM product, program, or service is not intended to state or imply that only IBM's product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any of IBM's intellectual property rights may be used instead of the IBM product, program, or service. Evaluation and verification of operation in conjunction with other products, except those expressly designated by IBM, is the user's responsibility.

APACHE INFORMATION. This information may include all or portions of information which IBM obtained under the terms and conditions of the Apache License Version 2.0, January 2004. The information may also consist of voluntary contributions made by many individuals to the Apache Software Foundation. For more information on the Apache Software Foundation, please see http://www.apache.org. You may obtain a copy of the Apache License at http://www.apache.org/licenses/LICENSE-2.0.

IBM may have patents or pending patent applications covering subject matter in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

> IBM Director of Intellectual Property & Licensing
> IBM Corporation
> North Castle Drive
> Armonk, NY 10504-1785
> USA

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

> IBM Corporation
> Mail Station P300
> 2455 South Road
> Poughkeepsie, NY 12601-5400
> USA
> Attention: Information Requests

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

Trademarks and service marks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both. If these and other IBM trademarked terms are marked on their first occurrence in this information with a trademark symbol (® or ™), these symbols indicate U.S. registered or common law trademarks owned by IBM at the time this information was published. Such trademarks may also be registered or common law trademarks in other countries. For a current list of IBM trademarks, visit the IBM Copyright and trademark information Web site (www.ibm.com/legal/copytrade.shtml).

Microsoft and Windows are trademarks of Microsoft Corporation in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

Other company, product, or service names may be trademarks or service marks of others.

Index